madcap software

USER GUIDE

MADCAP FLARE ONLINE

Authoring Guide

Copyright © 2026 MadCap Software. All rights reserved.

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of those agreements. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or any means electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of MadCap Software.

MadCap Software
1660 17th Street, Suite 201
Denver, Colorado 80202
858-320-0387
www.madcapsoftware.com

THIS PDF WAS CREATED USING MADCAP FLARE.

CONTENTS

CHAPTER 1

Introduction 5
- Permission Required? 6

CHAPTER 2

Adding Files 8
- How to Add a New File 8

CHAPTER 3

Uploading Files 13
- How to Upload a File 13

CHAPTER 4

Editing Files 14
- Single Authors - How to Edit an Existing File 15
- Content Editor Toolbar 19
- Code Toolbar 23
- Info Bar 25
- Copying and Pasting Content 29
- Breaks 31
- Markers 33
- File Properties 35

CHAPTER 5

Collaborative Authoring 38
General Information for Collaborative Authoring 41
Main Activities for Collaborative Authoring 49
Other Activities for Collaborative Authoring 59

APPENDIX

PDFs 85

CHAPTER 1

Introduction

MadCap Flare Online's cloud environment empowers you to author project files directly without having to use Flare Desktop. You can add new files, edit existing content, upload files, and even use integrated ChatGPT). The Flare Online interface includes an easy-to-use editor for content viewing and editing, and if permission is granted, you can also edit in a code editor.

- "Adding Files" on page 8
- "Uploading Files" on page 13
- "Editing Files" on page 14

☆ EXAMPLE Your documentation team has created a Help system using MadCap Flare Desktop. Your larger organization uses MadCap Flare Online as its platform to manage the content. A director, who is not trained in using Flare Desktop, views some content from the project in Flare Online and sees an ideal place to insert a relevant new topic. Without having to go through a review process or track down a writer to do the work, the director uses Flare Online to quickly create and add a topic to the project. The change is committed to the project. Anyone who works in the project in Flare Online will see the change, and Flare Desktop users will see updates once the remote and local repositories are synchronized.

Permission Required?

Editing content and project files is an activity available to users with the Author status. By default, users with Author status have the following permissions set:

- 

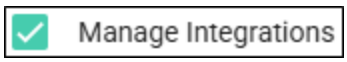
  If this is deselected, then viewing files in a read-only mode is allowed. On the left side of the page, the Files vertical three-dot menu is not available.

- 

  If this is deselected, the XHTML in the Code view is read-only.

  Editing code is regarded as a capability for an advanced user. If not done properly, the code can become malformed quickly. Administrators can prevent users from editing the code by deselecting the Edit Code permission.

In addition, AI Assist involves the following permissions:

- 

  This is required to integrate a ChatGPT account with a Flare Online license in the license settings.

- 

  This is required to use AI Assist (and therefore ChatGPT) when modifying topics and snippets.

  > NOTE Even if this permission is enabled, ChatGPT does not scan anything on your computer. The only information ChatGPT can acquire from you is what you enter manually into the prompt when using AI Assist. If your company has strict policies against AI or ChatGPT, simply do not use it.

For more information about permissions, see the Help system.

NOTE For the authoring feature to work properly, your project must be single-bound to Flare Online as the primary source control provider. The authoring feature does not support dual-bound projects.

NOTE If an author needs to work with the project in Flare Desktop after it is created in Flare Online—because advanced features are needed for the project—the user needs to (1) have access to the project, and (2) open Flare Desktop and import the project from Flare Online. If additional changes are made in Flare Desktop or Flare Online, the work would need to be synchronized between the local and remote repositories.

NOTE Since Flare Online is a remote repository, those who use Flare Desktop after changes are made in Flare Online, need to synchronize their remote and local repositories.

- Flare Online side Content is authored and committed to the project in Flare Online.
- Flare Desktop side To interact with updated content in Flare Desktop, use source control to pull changes from the remote repository and sync it to the local repository.

If two authors are editing the same file, at the same time, but one is working in Flare Desktop and the other is working in Flare Online, there is an auto-merge feature that detects external commits.

CHAPTER 2

Adding Files

From the Workspace page, you can add a new file to a project.

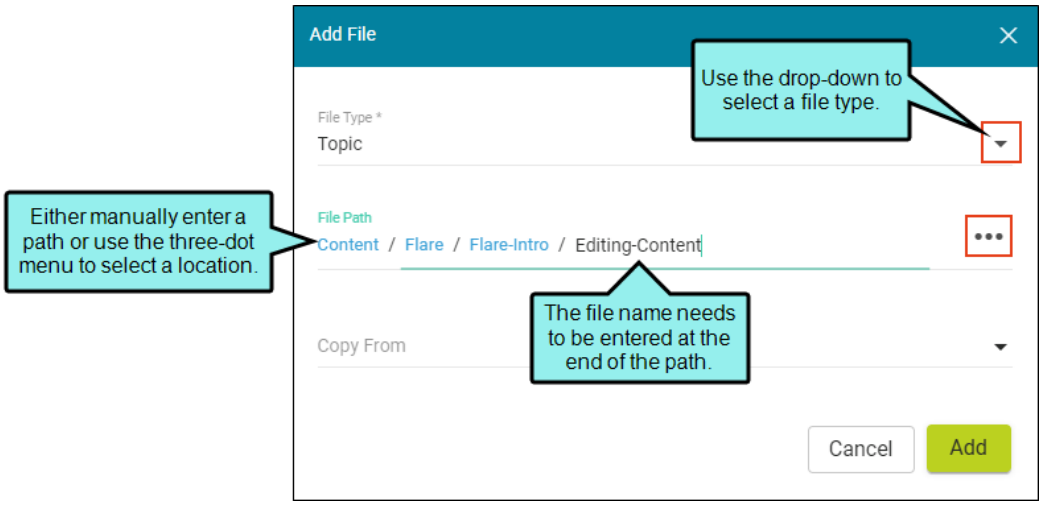
How to Add a New File

1. On the left side of the Flare Online interface, click **Projects**.
2. Click **Workspace** at the top of the screen.
3. From the main toolbar, click  to add a new file.
4. In the Add File dialog, from the **File Type** drop-down, select a required file type.
    - **Topic** Creates a topic with the file extension HTM, and it must be placed in a content folder.
    - **Snippet** Creates a snippet with the file extension FLSNP, and it must be placed in a content folder.
    - **Template Page** Creates a template page with the file extension FLMSP, and it must be placed in a content folder. The recommended path is Content > Resources > TemplatePages.
    - **Target** Creates a target with the file extension FLTAR, and it must be placed in a target folder.
    - **TOC** Creates a table of contents (TOC) with the file extension FLTOC, and it must be placed in the TOCs folder.

- **Variable Set** Creates a variable set with the file extension FLVAR, and it must be placed in a variable set folder.
- **Condition Tag Set** Creates a condition tag set with the file extension FLCTS, and it must be placed in a condition tag set folder.
- **Branding Stylesheet** Creates a branding stylesheet with the file extension CSS, and it must be placed in a content folder. The recommended path for branding stylesheets is Content > Resources > Branding.
- **Stylesheet** Creates a regular stylesheet with the file extension CSS. The recommended path for stylesheets is Content > Resources > Stylesheets. You can only edit this file type in Code mode of the workspace, but you can apply CSS styles from your stylesheet in the Content Editor.
- **Destination** Creates a destination file with the file extension FLDES. This defines the folder(s) and file path where you want your publication to go for whichever host (i.e., connector) you are using.
- **Skin** Creates a skin to your project with the file extension FLSKN. It can be based on an existing file or on a factory template (i.e., HTML5 Side Navigation and HTML5 Top Navigation).
- **Other** Creates a text-based file (e.g., TXT file). With this file an extension is not necessary, and it can be placed anywhere in the project.

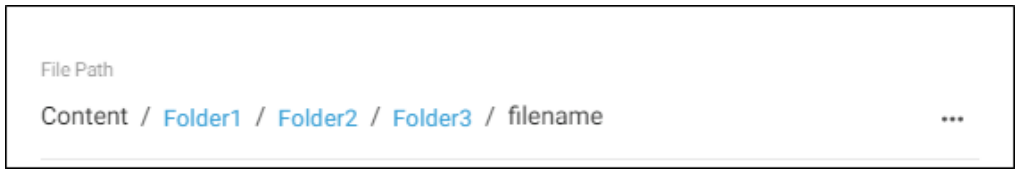
5. In the **File Path** field, enter a path and a name for the new file.

   Alternatively, click ••• to select a location for the file in the project, and **Accept** the file path. Then in the **File Path** field, enter a name for the file.



NOTE You might notice the File Path displays black or blue lettering. The blue items are folders, and you can click the item to jump to that folder.

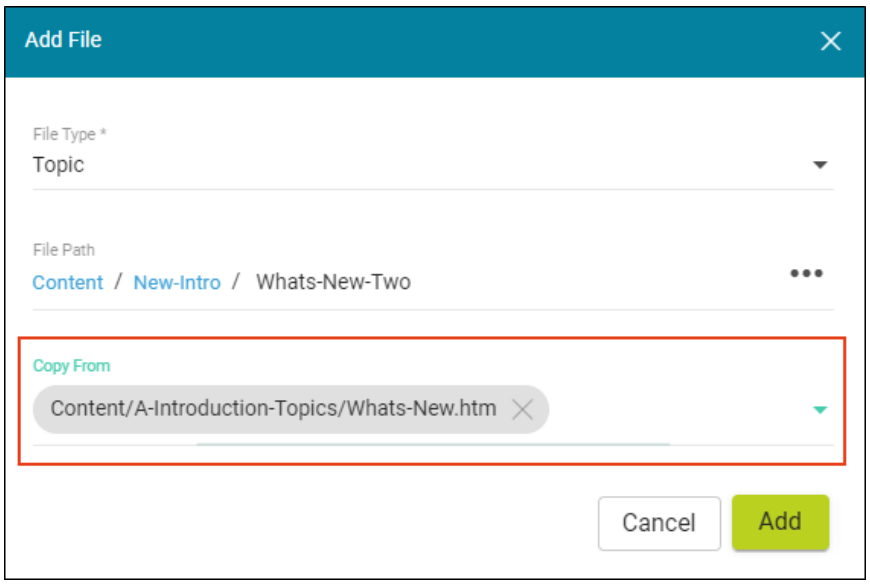
For example, in the Add File dialog, a file path contains several folders (indicated by blue lettering) before the filename.



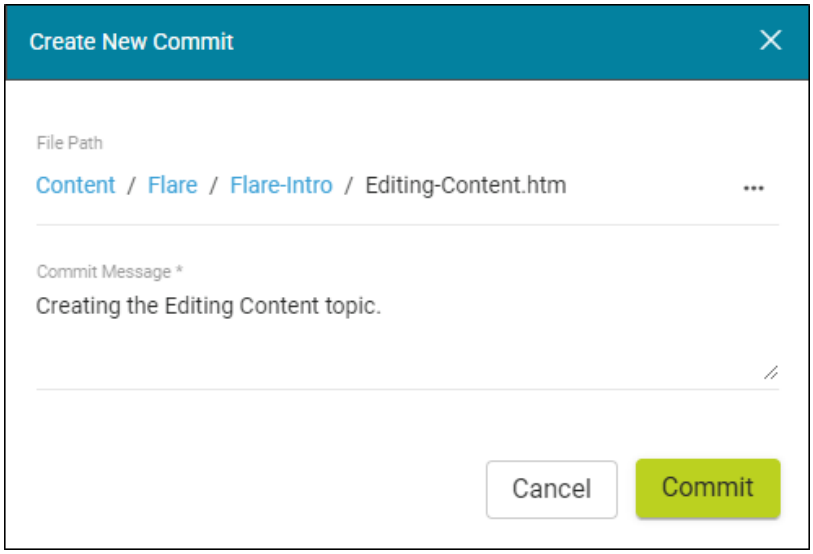
If the "Folder1" item is clicked, you know longer see the subfolders.



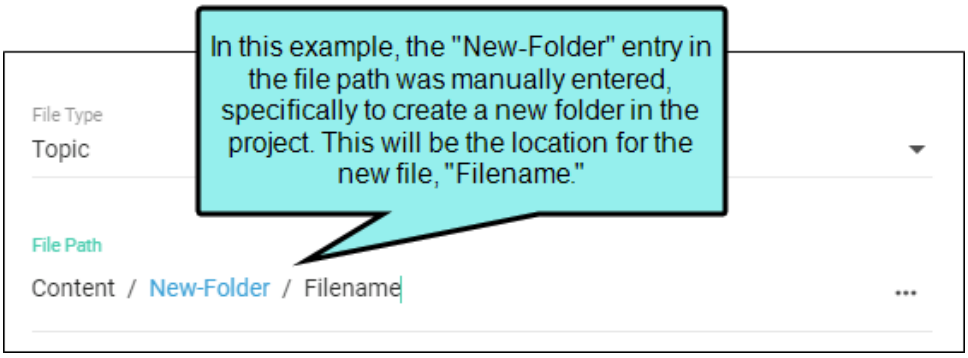
6. (Optional) In the Add File dialog, from the **Copy From** drop-down, you can select an existing file to use. This copies all the content or settings from the existing file to the new file, providing a base of information to start with for your file. If you choose not to select a file here, the new file will be based on factory content and settings.



7. Click **Add**. The new file opens in the editor to the right of the project files.
8. Make changes in the editor, and depending on the type of file, you can also use the toolbar to manage the content.
9. In the upper-right corner of the editor, click **Commit**.
10. In the Create New Commit dialog:
    a. Confirm the **File Path** (or enter a different path).
    b. Type a **Commit Message**.
    c. Select **Commit**. The new file displays in the project files.



TIP What about creating a folder? Since Flare Online is Git-based, you can only create a new folder by creating a file. Git does not allow empty folders. One way to get around this is to create a folder in the file path when adding a new file type. Just type the name of the new folder, followed by a slash.



Once the new folder and file are added, they display with the other folders and files for the project.

NOTE When you create a new file, it creates a file in a "pending add" state, and the file displays in the Files tree. You can add as much content to the new file as you like initially, and other authors can make edits to it as well. The timeline updates as edits are made, but the file must be committed to the repository in order for it to show a version history.

CHAPTER 3

Uploading Files

Different types of files can be uploaded to the project's repository and placed in the file tree. Some types (e.g., topics, snippets) can be viewed and edited in the editor, while others (e.g., images) can only be viewed.

How to Upload a File

1. On the left side of the Flare Online interface, click **Projects**.
2. Click **Workspace** at the top of the screen.
3. From the main toolbar, click  to upload a file. The Upload File dialog opens.
4. Do one of the following:
    - In the **Folder Path** field, type a path to upload files from.
    - From your local computer, drag and drop files to the Upload File dialog.
    - Click **Browse for files** and interact with the Open dialog to navigate to the files you want to upload.
5. In the **Commit Message** field, enter a (required) comment for committing the file(s).
6. Click **Commit**.

CHAPTER 4

Editing Files

Flare Online offers a collaborative authoring workspace. You can work as a single user in the workspace, or a team of many authors can just as easily use the workspace. Multiple authors can edit collaboratively and concurrently (i.e., the same file at the same time) and see everyone's updates in real-time without the worry of losing work or running into conflicts. See "Collaborative Authoring" on page 38.

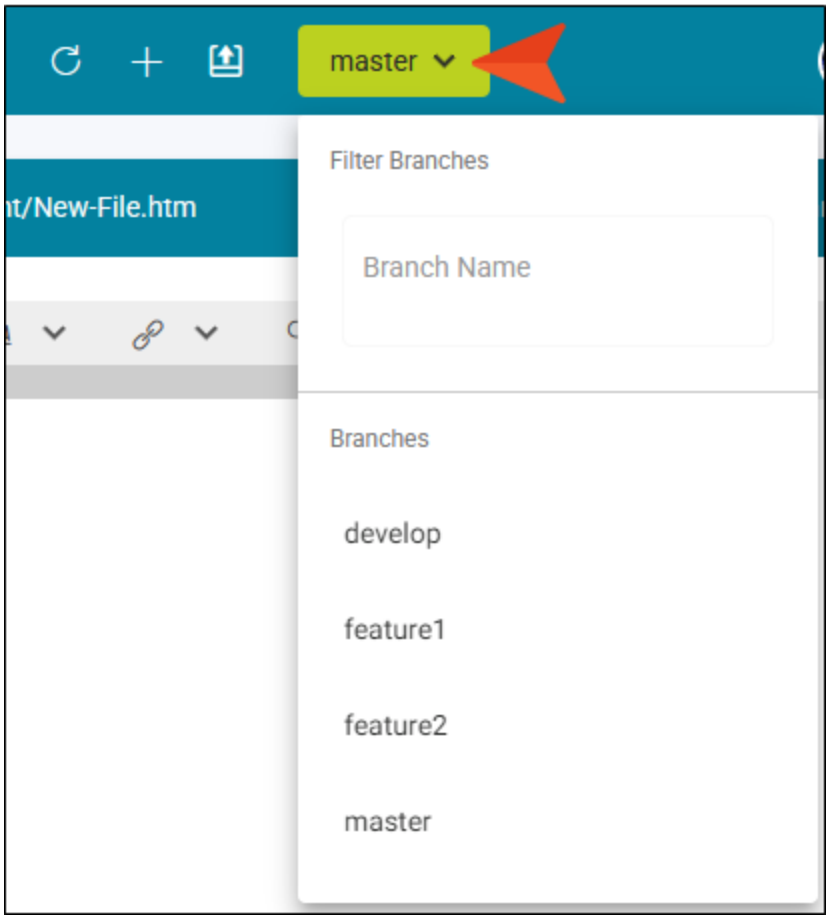
Content files (e.g., topic, snippet) and some project files (e.g., target, condition tag set, variable set) can be edited in Flare Online. By default, the workspace displays a file in Content mode, which is an easy-to-use light-weight editor. You can also display a file in Code mode, but this is for advanced users who prefer to edit in XML code directly. Both modes are equipped with basic editing tools. Some file types, such as page layouts, can be opened in Code mode only.

This chapter discusses the following:

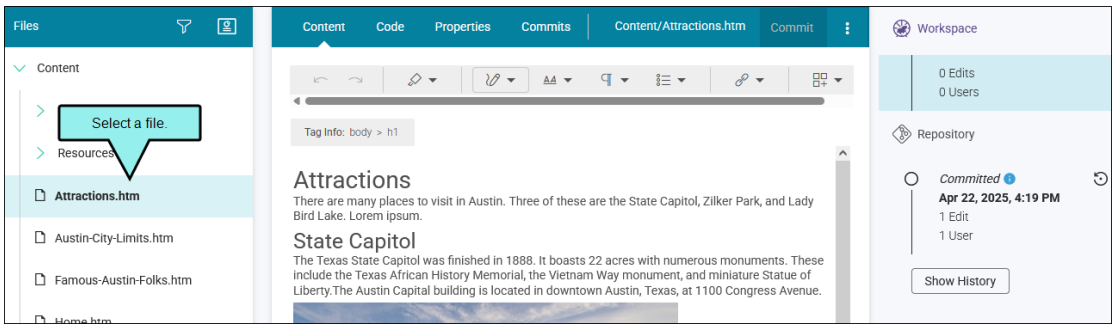
Single Authors - How to Edit an Existing File 15
Content Editor Toolbar 19
Code Toolbar 23
Info Bar 25
Copying and Pasting Content 29
Breaks 31
Markers 33
File Properties 35

Single Authors - How to Edit an Existing File

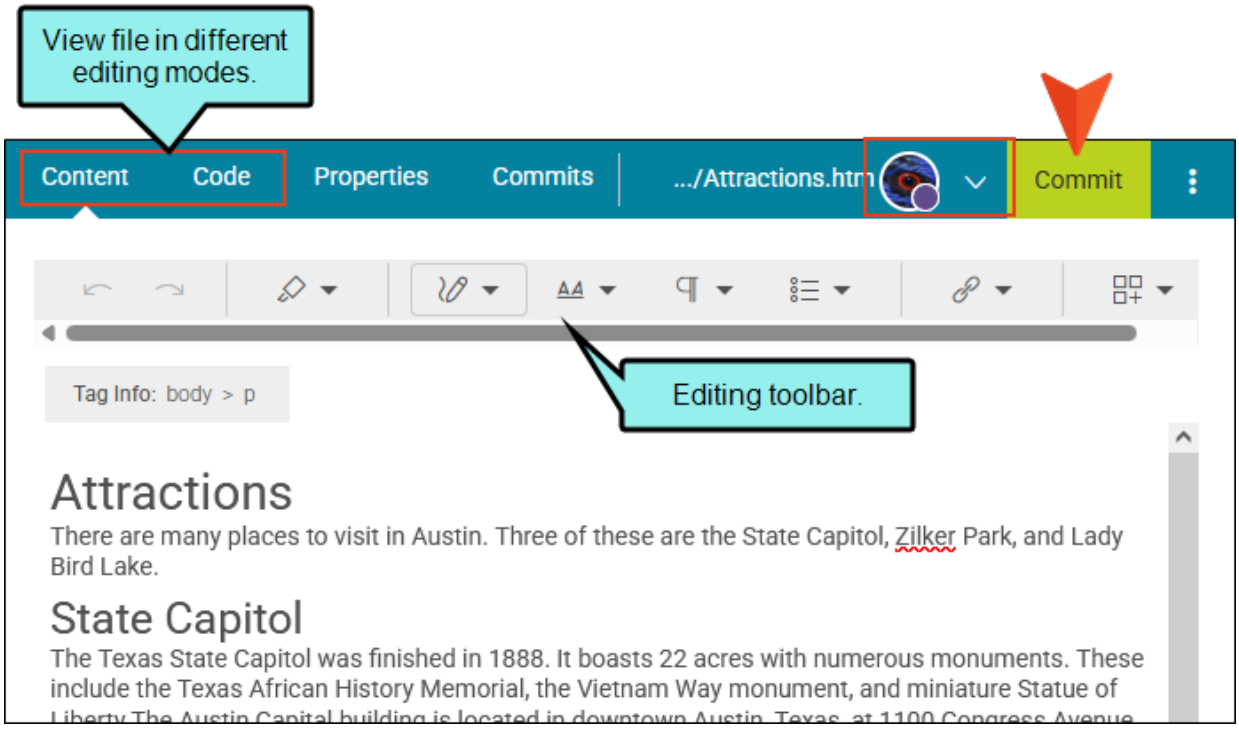
1. On the left side of the Flare Online interface, click **Projects**.
2. Select a project to open it.
3. Click the **Workspace** tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.



5. (Optional) Click . The Workspace Overview opens to display various items (e.g., status, authors, type) about edited files in a project. (If files are already in a working state, the Open Workspace Overview button shows with a circle in the upper-right corner.)
6. From the left side of the page, expand the existing folders to navigate to a file. You can also click to search for a specific file.
7. Select a file. It displays in the editor to the right. The right gutter switches from showing project activities to displaying a version history for the file.



8. Click in the workspace and make an edit (this activates the file in an uncommitted state for editing, auto-sets your status to In Progress, enables the Commit button, and begins to populate the version history). Use the toolbar at the top of the editor to manage the content. Also, the tabs at the top of the editor (i.e., Content, Code, Properties, and Commits), allow you to switch modes so you can edit the content, markup, or properties, and view commit details.



> NOTE You can toggle between the editing modes at any time. If you switch and then close or commit a file, then the next time you open the editor, it opens to the last mode you left off.

9. When you are done editing a file, it needs to be committed into the repository. Click **Commit**.

> NOTE From your avatar in the editor's toolbar, you have the option to select Ready to Commit, but you do not have to since you are the only author editing. You can just click Commit when you are done.

10. In the Create New Commit dialog, verify the new file path, and enter a **Commit Message**.

11. Click **Commit**. (Or click **Cancel**, to back out of the process.)

NOTE Keep the following in mind when loading files:

- The file size limit is 5 MB. This limitation helps prevent issues.
- If changes approach the limit, a message warns you.
- If edits exceed the size limit (or if the file starts out already over the limit), an error displays. If you are on the Content tab, the error is triggered when you attempt to commit the changes or switch to the Code tab.
- If you are on the Content tab and see the error message, you can make edits to reduce the file size and then commit the changes.
- If you are on the Code tab and see the error message, you can still edit the file and commit the changes, but you cannot switch to the Content tab.

Content Editor Toolbar

You can use the options in the toolbar at the top of the Content Editor to accomplish different tasks.



Option	Description
	Reverses the most recent action.
	Repeats the most recent action that you reversed.
	Tracks changes in the Content Editor. When toggled on, changes made to content are highlighted in a colored font, and a track changes rectangle is added to a sidebar to the right. For example the rectangle indicates if content is added or deleted. Adds an annotation rectangle, extended to a sidebar to the right, with a line pointing to the selected content. This rectangle will hold annotation text after you type it. Also, the selected content is shaded, indicating that the annotation refers to that text. However, the shading will not be displayed in the output; it is for internal use only. Removes the selected annotation from the document.
	Displays the active CSS style and all style classes for the current style tag, and other style tags that you can change to. The drop-down is dynamic to the content selected, and is valid for block-level and character-level content.
	Applies bold typeface to the content selected in the topic. Applies italic typeface to the content selected in the topic. Underlines the content selected in the topic.

Option	Description
	Lets you apply basic styles to block-level content: ■ Paragraph Applies a paragraph tag to the content. ■ Heading 1 - Heading 6 Applies a heading tag (<h1> through <h6>) to the content.
	Lets you create and set list styles on content:  Applies a bulleted list tag to the content.  Applies a numbered list tag to the content.  Applies definition list tags to the content.  Outdents the list item(s). This option pertains only to lists, not other kinds of content.  Indents the list item(s). This option pertains only to lists, not other kinds of content.
	 Lets you insert or edit a text hyperlink in a content file.  Removes a hyperlink, leaving the text in place.  Lets you insert or edit a cross-reference in a content file (e.g., topic, snippet).  Removes a cross-reference link, leaving the text in place.

Option	Description
	 Opens the Image dialog. Use to locate an image file, add alternate text, and set width and height attributes of the image to put in the editor.  Opens the Insert Multimedia dialog. You can locate a multimedia element (such as a video) in the project and insert it into the Content Editor. You can enter the URL for a web video, such as YouTube.  Opens a dialog that lets you select a variable set on the left and then choose a specific variable on the right to insert into the file. This is available when authoring and editing files, not for topic reviews.  Opens a dialog that lets you navigate to and select a snippet on the left. This is available when authoring and editing files, not for topic reviews. There are two types of snippets: text and block. This is determined by the way you insert the snippet.  Displays a small rectangle with a keyboard icon, as well as a full mathematical browser keyboard at the bottom of the workspace. You can type directly in the rectangle or use the full keyboard to create equations. There is also a small icon in the rectangle to choose various equation options.  Converts the selected content to a drop-down, with the first line serving as the hotspot and the content below as the body. In the output, users can click the hotspot to expand and collapse the body content. This is available when authoring and editing files, not for topic reviews.  Select from the drop-down a proxy to insert or edit. It opens a dialog the proxy where you can interact with proxy properties.
	Opens a drop-down that lets you select the number of columns and rows that you want to add as you insert a new table.

Option	Description
	Opens AI Assist, which lets you converse with ChatGPT. You can then insert or replace content with ChatGPT's response. This is available when authoring and editing files, not for topic reviews.
	Opens a dialog that lets you apply conditions to content.
	 Shows or hides colored shading (and in some cases a colored square) where conditions have been applied.  Shows or hides styles rendered from the cascading stylesheet.

NOTE Flare Online's lightweight editor is responsive. This means that elements of the workspace—local toolbar items, shift automatically if you resize the UI, move from a large screen to a smaller one, or change the resolution on your monitor.

Code Toolbar

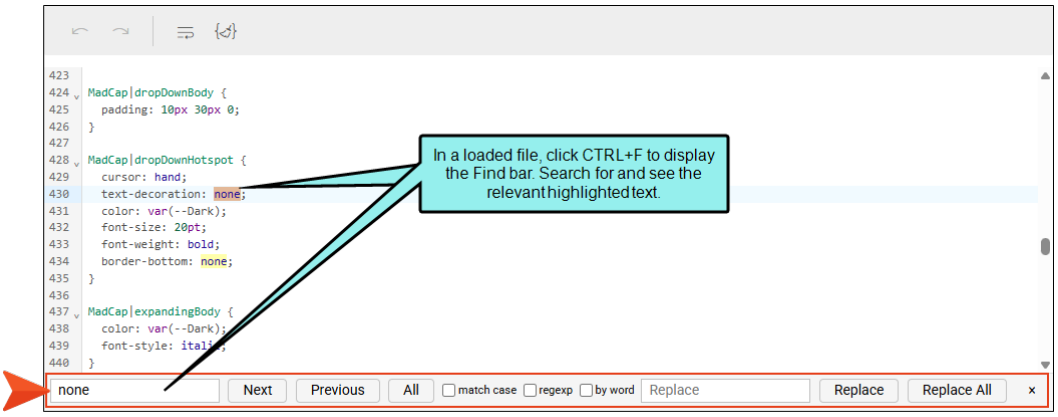
You can use the options in the toolbar at the top of the Code Editor to accomplish different tasks.



Option	Description
	Reverses the most recent action.
	Repeats the most recent action that you reversed.
	Toggle to break a section of text so that it fits into the display area of the Code Editor. Toggle again for no wrap to display.
	Formatting the code cleans up the white space in the code. This can be performed on HTML, CSS, and JS file types.

TIP When you are using the Code Editor, you can click CTRL+F to search for content with a Find bar. This is convenient if a loaded file has hundreds of lines of code and you do not want to scroll through all of it for specific information. (This is not available in the Content Editor.)

1. Open a file in the Code Editor.
2. Click in the editor to give it focus.
3. From your keyboard, click **CTRL+F**. (Notice the Find bar displays at the bottom.)
4. In the Find box, enter text or data to locate. You can search for text patterns (e.g., match case, regexp) or replace text.



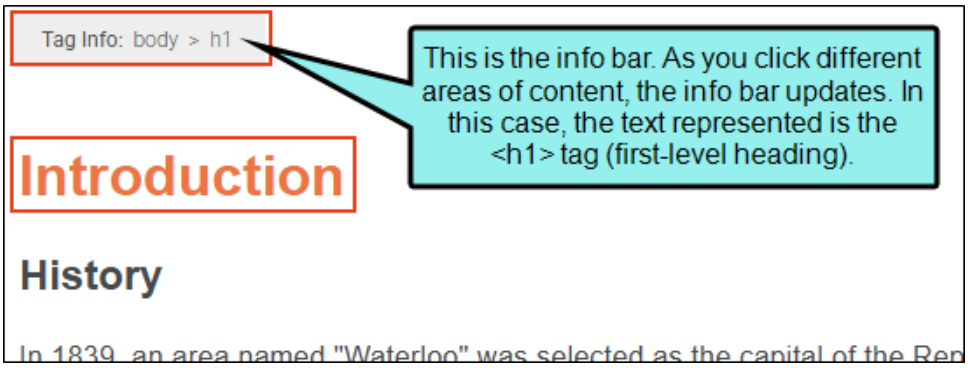
Info Bar

Many different kinds of elements (e.g., tags, annotations, markers) can display in the editor as it is being authored or edited. The Content Editor provides an info bar just under the toolbar. This bar displays details and is clickable in some cases.

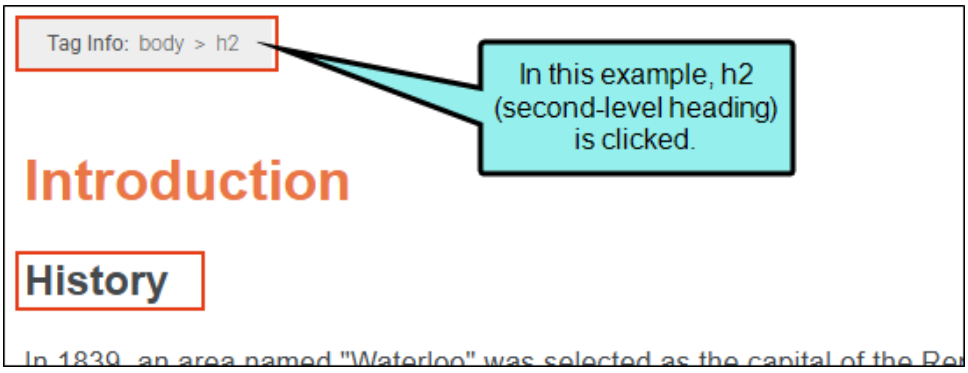
Display Details

When you click in the editor, this info bar lets you know what that area contains. It also displays information such as the tag structure, the destination of a link, or annotation details.

☆ EXAMPLE If you open a topic for editing, notice the info bar near the top of the top of the Content Editor.



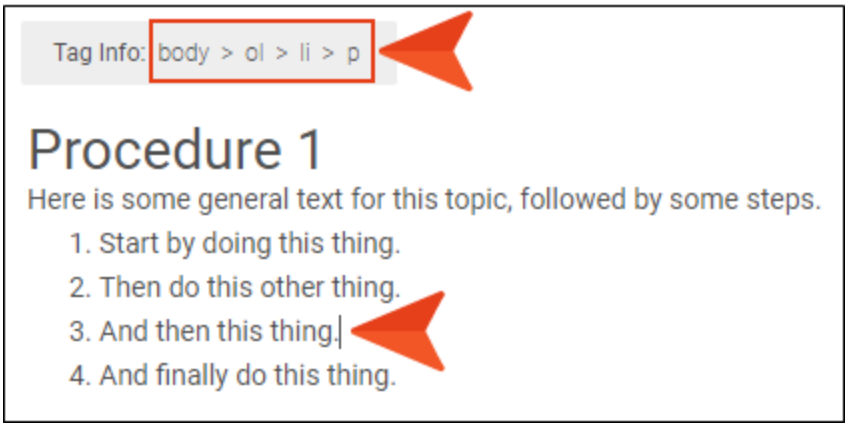
If you click on a different area, the info bar updates.



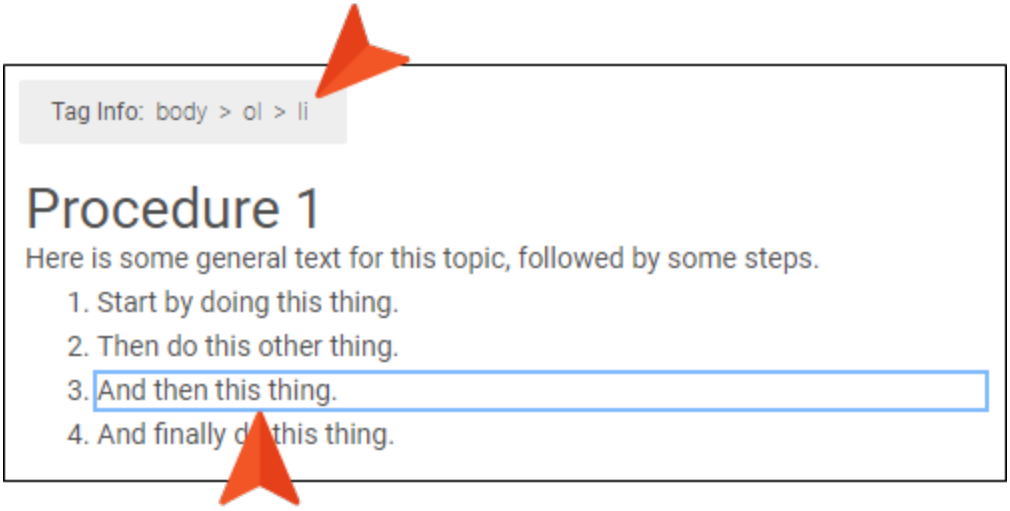
Clickable Tags

You can click any tag in the info bar, and the corresponding area in the content file will be selected as well.

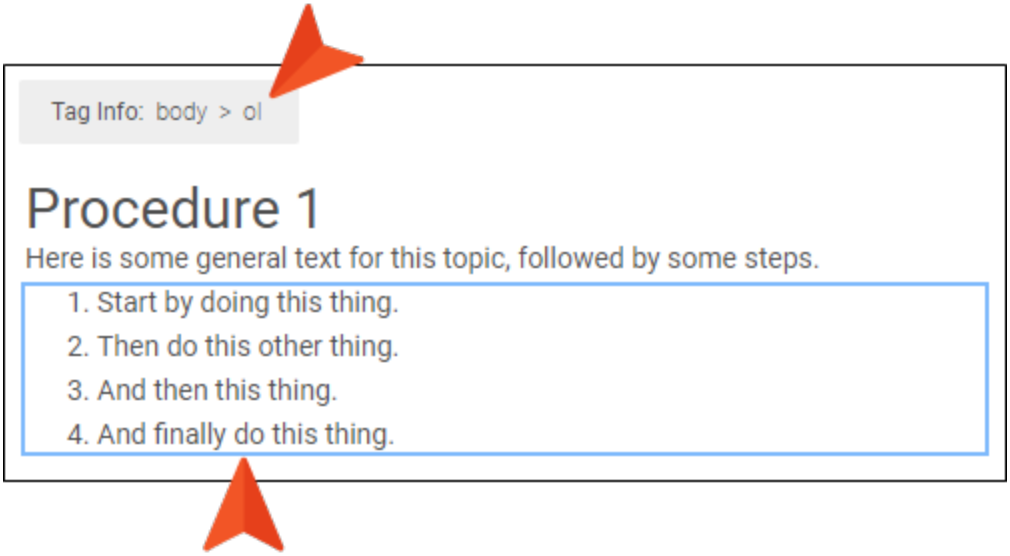
☆ EXAMPLE You have a topic with a numbered list, and the cursor is currently located within it, at the end of step 3. Notice that the info bar describes the tag structure. The broadest tag is <body>, which essentially holds all of the content in the file. This is followed by <ol> (which is an ordered, or numbered, list), since the cursor is located within that list. Within that tag is <li>, which represents the list item (in this case, the third list item). And finally, within that list item is a paragraph (or <p> tag).



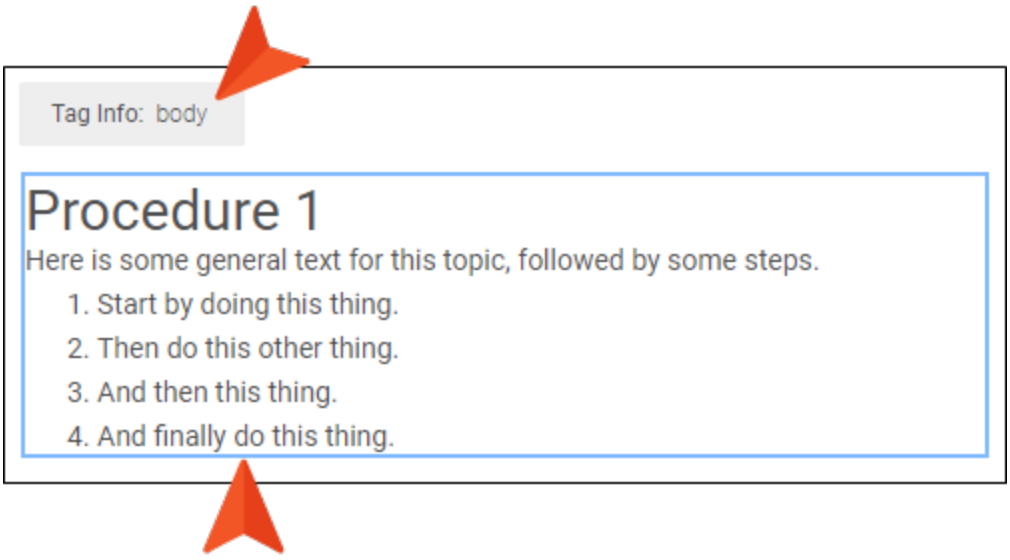
If you were to click li in the info bar, the list item would be selected. (Clicking the p tag would look much the same in the editor.)



If you click ol in the info bar, the entire list area would be selected.



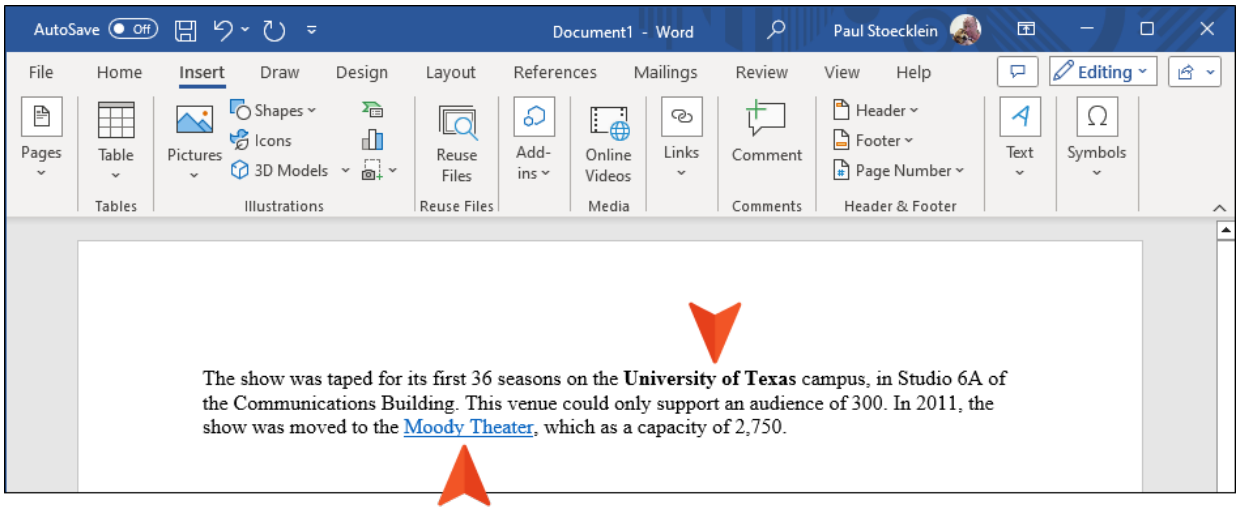
☆ And finally, if you click **body**, everything in the file would be selected.

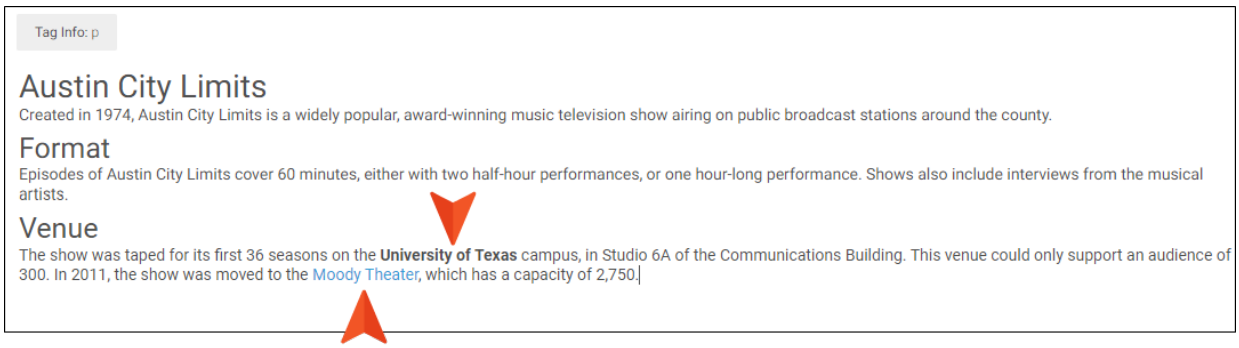


Copying and Pasting Content

Keep the following in mind when copying and pasting content:

- **Text** When copying and pasting text, formatting is retained. This includes lists, hyperlinks, and other styles (e.g., bold, italics, underline), even when the text is copied from applications outside Flare Online.

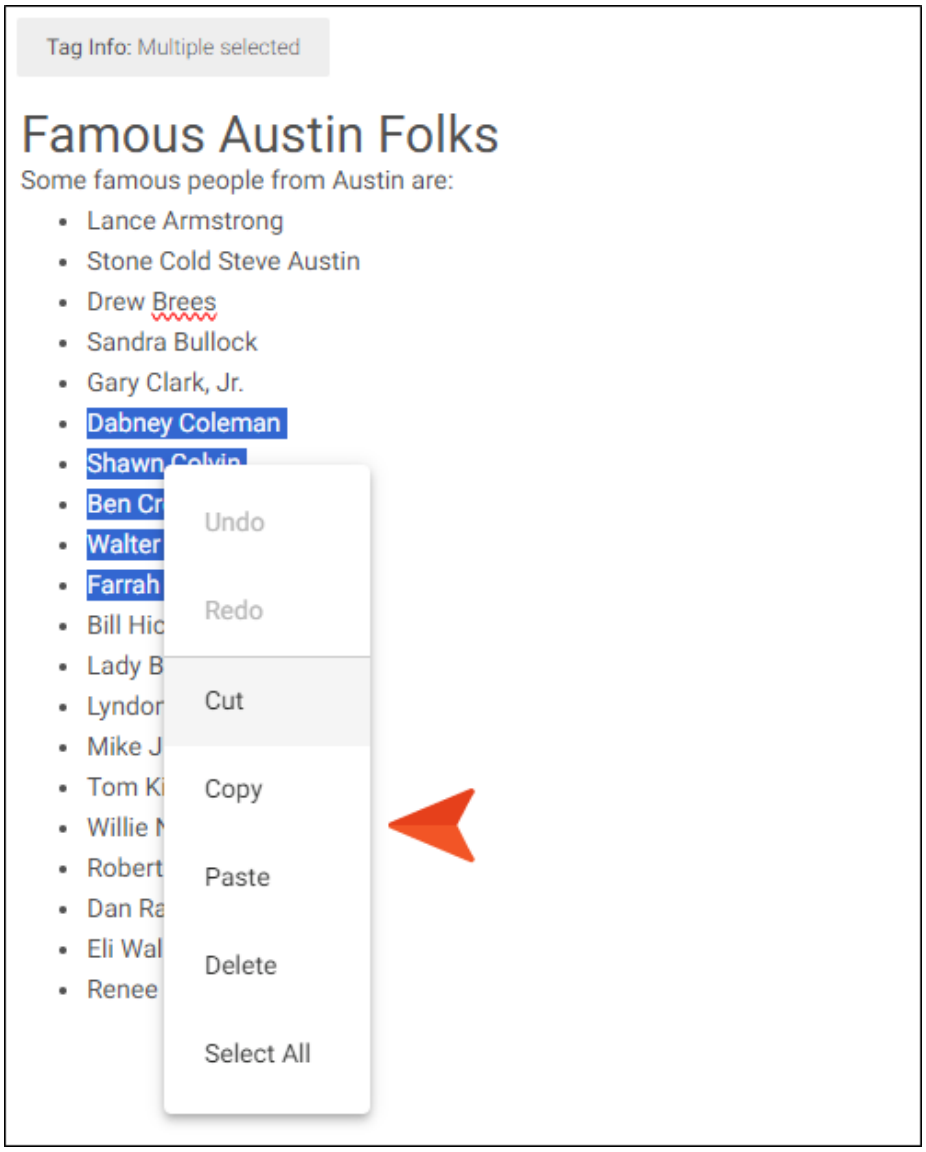




- **Tables** Formatting for tables is retained when copying and pasting within and between files in Flare Online. Pasting tables from other software (e.g., Word, Outlook) will be inserted as simple tables.
- **Images** You can also copy and paste images, but only when you are doing so within the same topic or snippet. It does not work from outside sources.

When copying and pasting (and performing other actions, such as cut, select all, and undo), you can use standard shortcuts (e.g., CTRL+C, CTRL+V) on your keyboard.

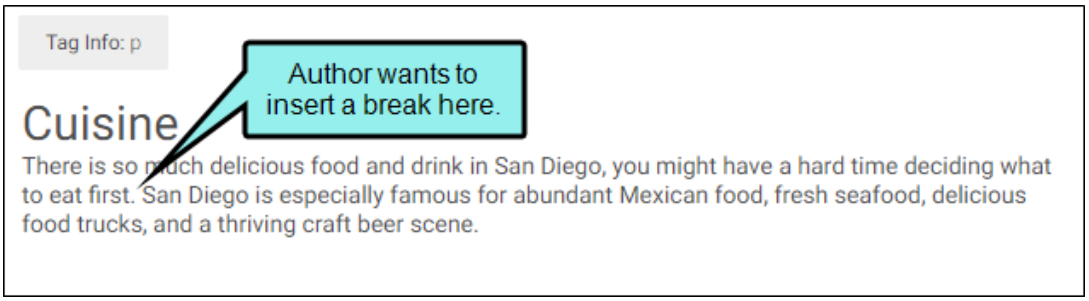
You can also right-click and select from a menu.



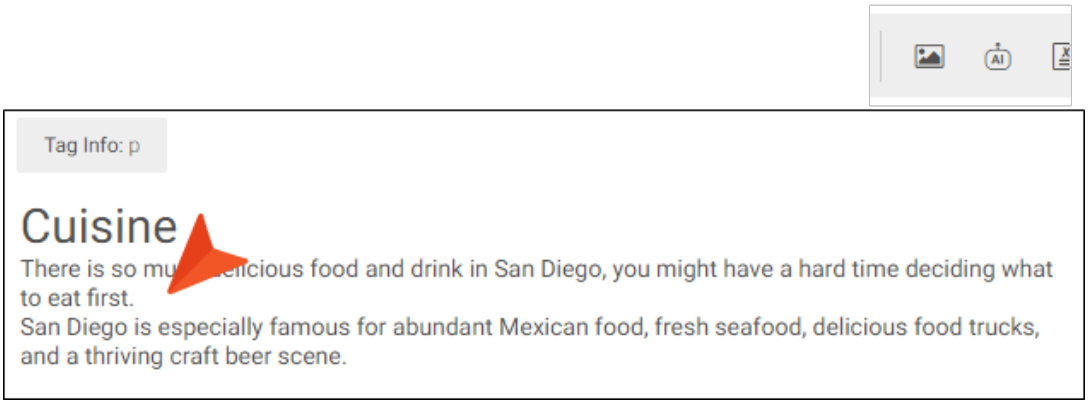
Breaks

You can insert break tags by pressing SHIFT + ENTER on your keyboard.

☆ EXAMPLE You have a topic open in Flare Online's editor. The text looks like this, but the author wants to insert a break mid-paragraph.



After pressing SHIFT + ENTER, the Content Editor shows it as a break in the line of text.



☆ You can use the Code Editor to view breaks. It displays with the <br /> tag.



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <html xmlns:MadCap="http://www.madcapsoftware.com/Schemas/MadCap.xsd">
3
4  <head>
5  </head>
6
7  <body>
8    <h1>Cuisine</h1>
9    <p>There is so much delicious food and drink in San Diego, you might have a hard
   time deciding what to eat first. <br />San Diego is especially famous for abundant
   Mexican food, fresh seafood, delicious food trucks, and a thriving craft beer scene.
   </p>
10 </body>
11
12 </html>
```

Markers

Styles render well in the editor. However, some tags and markup represented by markers are necessary for topics and snippets. These provide a visual cue that more than plain text is present. In some cases, the author will not need to or be allowed to update the element. It is important to know that one of these elements is present, so it does not get deleted accidentally.

Marker	Elements
Banner 	Bookmarks
Dashed Border 	Conditions
Dotted Border 	Micro Content
Gray Box  Code not rendered will most likely display in gray boxes.	3D Models Concept Links Concepts Index Keywords Keyword Links
	Multimedia Page Breaks Proxies Related Topic Links Scripts

Marker	Elements
Solid Border 	Snippets Variables Page Footer Page Header

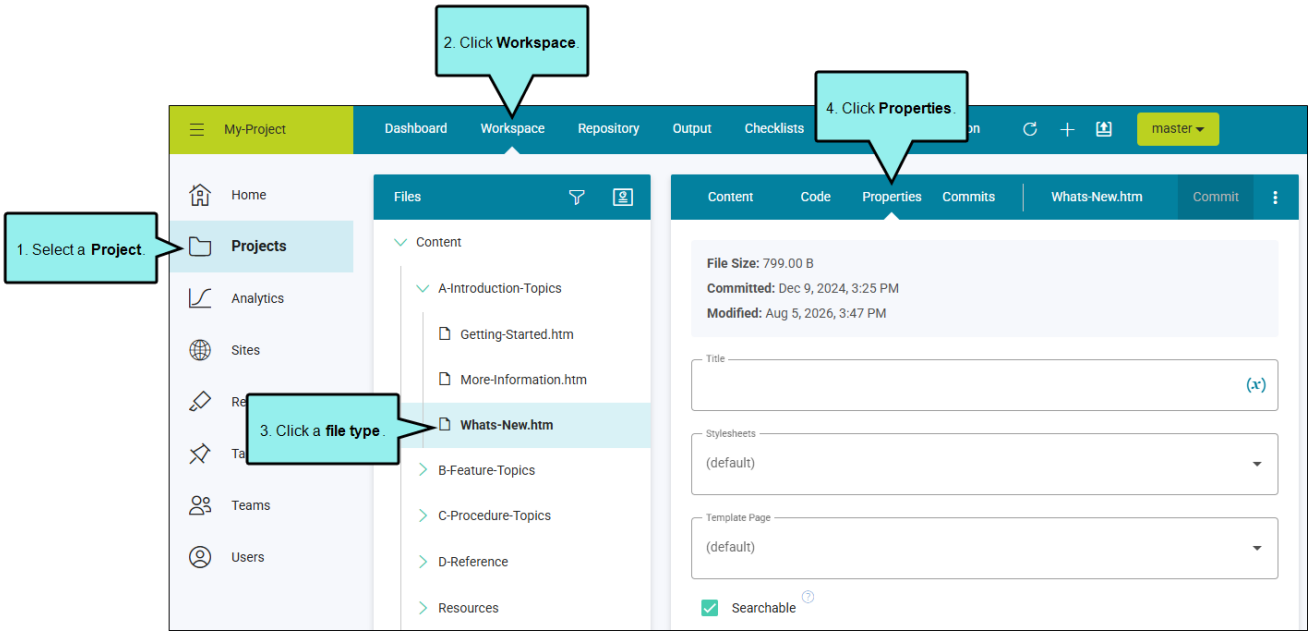
NOTE Some elements, such as eLearning items, cannot be created using the Content Editor or Code Editor. But, they can be viewed, and in some cases supported, for editing within the elements established by Flare Desktop.

File Properties

Flare Online lets you edit important file properties in a File Properties Editor. You can make quick file-level configuration changes during day-to-day authoring, cleanup work, or release preparation, and manage common file-property settings such as stylesheets, template pages, and whether a topic is searchable in published output. You can also view general properties for supporting assets like images and snippets.

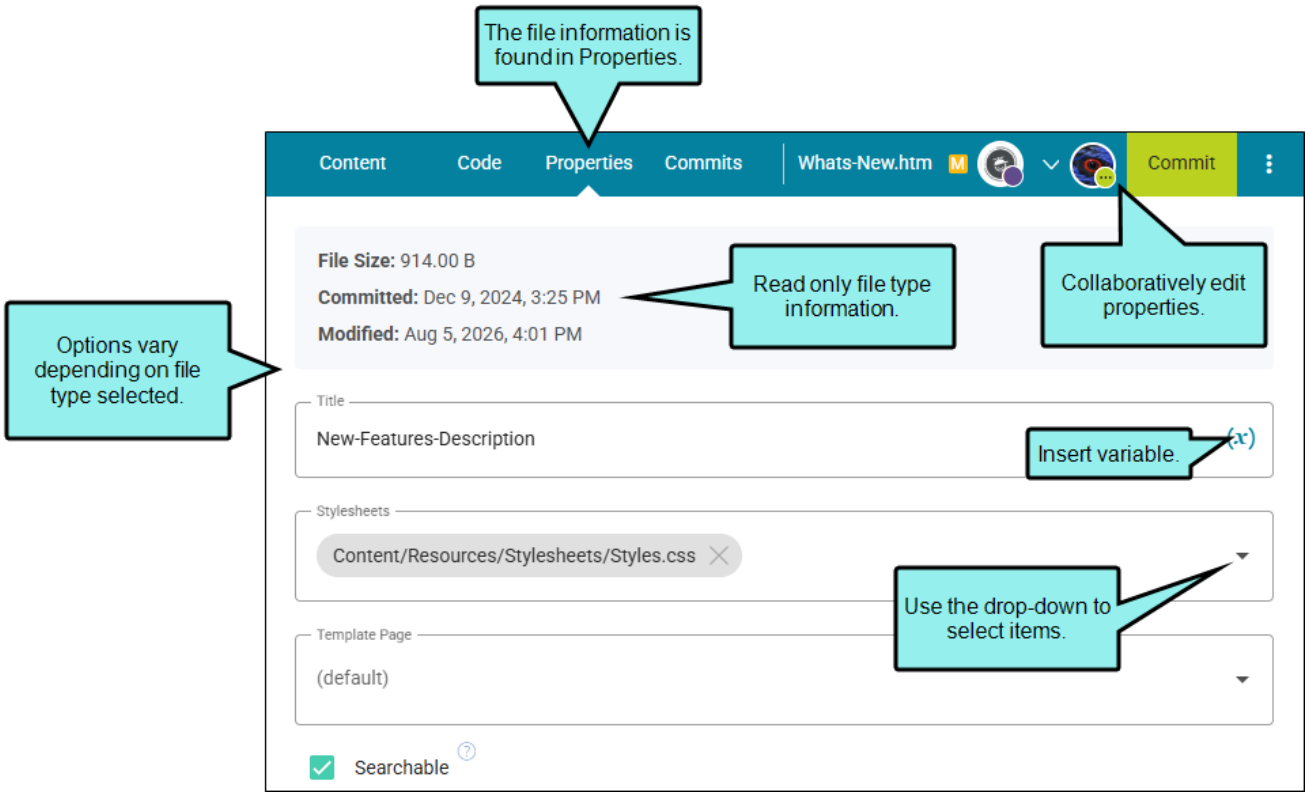
Where Are File Properties?

When you select a file, it displays in the workspace.



A Properties tab displays in the editor if applicable, with other relevant file tabs (i.e., Content, Code, Commits). Flare Online supports property information for all file types from the Content and Project folders but note that properties are based on the file type, not on the folder where the file lives. The File Properties Editor includes read-only information and various other options, depending on the file type. If the selected file has an editor or "viewer" (e.g., for an image), it opens the specific editor for that file by default.

In addition, multiple users can make collaborative edits to properties and retain all sets of updates when done.



NOTE The profile holds project file information and settings that you can apply to a project (i.e., icon color, description, initials, name), and the project file from the workspace lets you change the properties. A key difference is where the information is written. The profile writes to the database, whereas the workspace writes to the file and is tied to Git.

How to Edit File Properties

1. On the left side of the Flare Online interface, click **Projects**.
2. Select a project to open it.
3. Click the **Workspace** tab at the top of the screen.

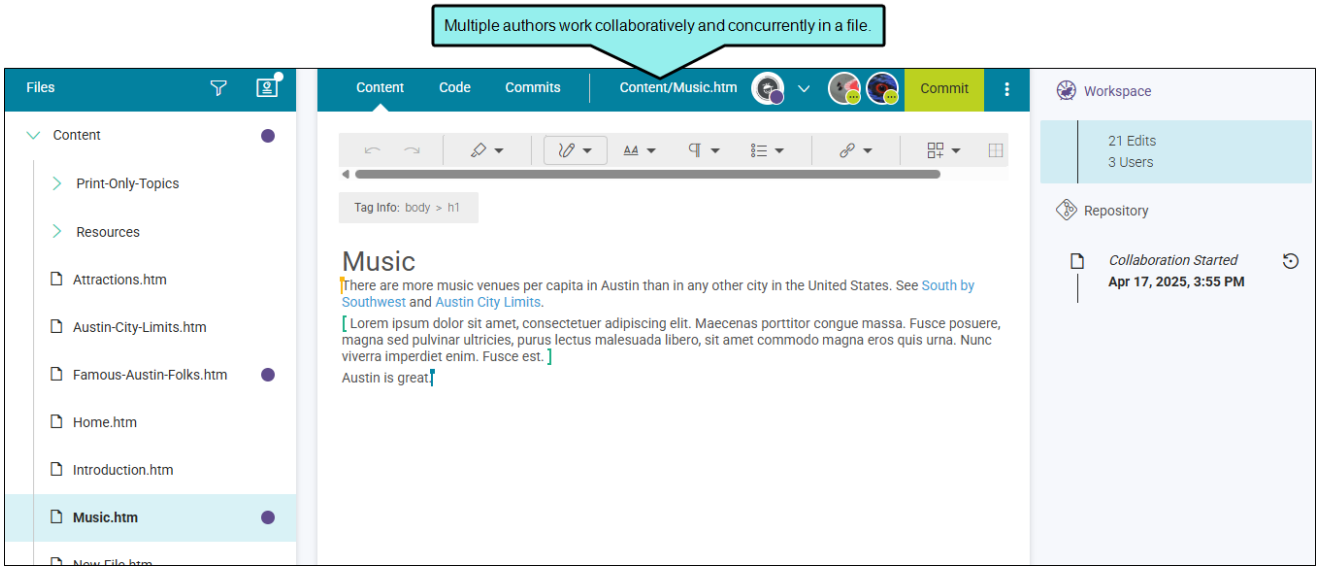
4. From the left side of the page, expand the existing folders to navigate to a file. You can also click  to search for a specific file.
5. Select a file. It displays in the editor to the right.
6. At the top of the editor, select the **Properties** tab.
7. Review properties or click to edit available properties.
    - **Title** Enter a title for the file (e.g., identify file purpose).
    - **Committed** (read-only) Date of last file commit to the Git repository (for all files).
    - **Modified** (read-only) Date of last document edited (only for collaboratively managed files).
    - **File Size** (read-only) The size of a file; not for folders.
    - **Dimensions** (read-only) Images only.
    - **Resolution** (read-only) Images only. Only shown for image types that hold DPI metadata, such as PNG, JPEG, BMP, and TIFF. GIF and SVG are treated as resolution-independent and never show this property.
    - **Stylesheet** Select from the drop-down to update a topic to use a different stylesheet.
    - **Template Page** Select from the drop-down to update a topic to use a different template page.
    - **Searchable** Select to mark a topic as searchable, to control how it displays in published output. Deselect to make a file unsearchable.
8. Click **Commit**.
9. In the Create New Commit dialog, enter a **Commit Message**.
10. Click **Commit**. (Or click **Cancel**, to back out of the process.)

 NOTE Edits made to a file's properties cannot be "undone" in the workspace history and are ignored by the track changes feature.

CHAPTER 5

Collaborative Authoring

Authors can work together in one centralized place to create, edit, and revise content in real-time using the collaborative authoring feature. This co-authoring allows for multiple authors to work on a document simultaneously (i.e., the same file at the same time), provides awareness of changes from other authors without the worry of losing work or running into conflicts, and promotes work management such as viewing status of who's working or what's been completed and version history.



General Information

- "Collaborative Editing Workspace" on page 41
- "Visible Changes From Others" on page 45
- "Do I Have to Worry About Conflicts?" on page 48

Main Activities

- "Editing With Multiple Authors" on page 49
- "Committing Edits" on page 54
- "Selecting Ready to Commit" on page 57

Other Activities

- "Discard Changes From the Workspace" on page 59
- "Using Version History" on page 62
- "Workspace Overview" on page 71
- "Merging Files With External Commits" on page 80

NOTE What is the repository versus the workspace? Flare Online stores files virtually in a Git-based system. If you open and view a file, it is in a committed state in the project's repository. If you click to edit a file, it then enters a "workspace mode," essentially making a working copy of the file from the repository. In this mode, the file is in a non-committed state with pending changes. When editing is completed, the file needs to be committed back to the project's repository.

NOTE What if I don't want others involved in certain files? If you plan to make changes in files, but would prefer that other authors do not make changes in those same files, there is currently no way to lock files in collaborative authoring. The best solution is to create another branch in Flare Desktop and work in that branch by yourself. You might need to ask other authors not to work in that same branch. Then later, when you are finished with all changes, you can merge that branch into another using Flare Desktop.

NOTE Certain types of files (e.g., readme, text) are single-commit items and exist in the repository only. Conversely, HTM files (e.g., topics, snippets, templates), some XML editors or non-HTML project files (e.g., condition tags, variables, TOCs, targets), and CSS file types are supported by collaborative authoring. You can open multiple files in the workspace, retain all changes, and then commit them in bulk.

General Information for Collaborative Authoring

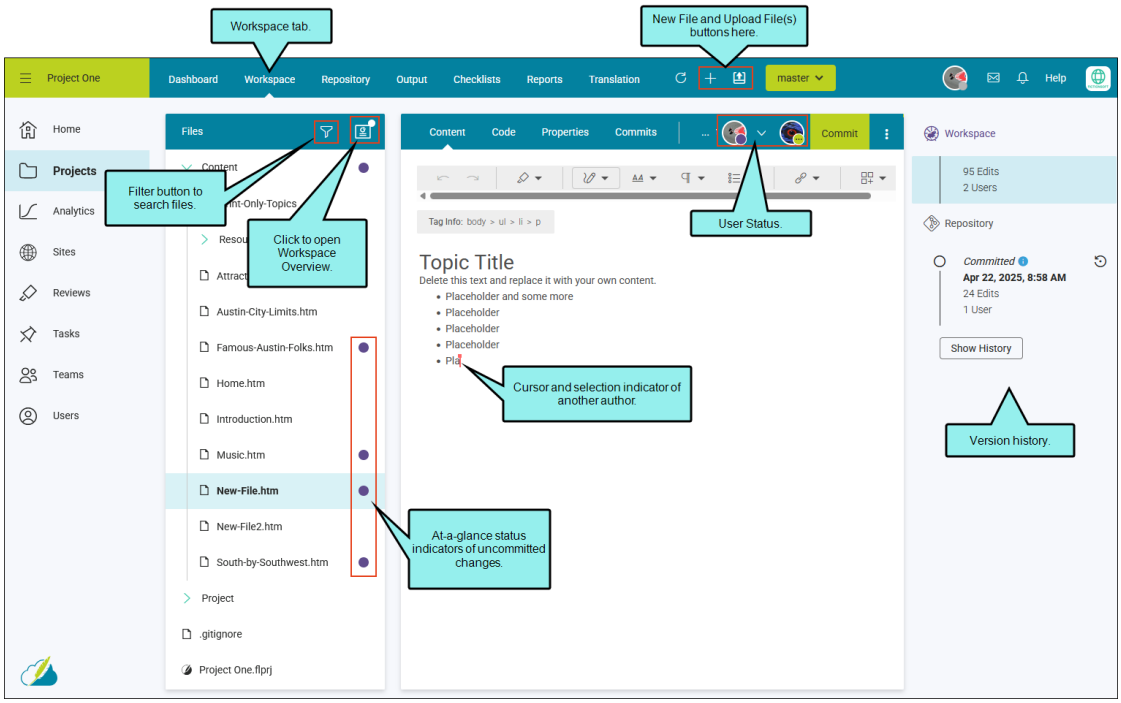
There are various pieces of general information you should know if you plan to use this feature.

Collaborative Editing Workspace

Flare Online co-authoring happens in the Content Editor. All authors work collaboratively in files, with full transparency of edits from everyone in that project. Regardless of whether you are a single author or a team of authors, you will use this workspace for writing and editing files.

Workspace Diagram

When you open a project, click the **Workspace** tab from the main toolbar. The elements found on the Workspace page let you edit content but also provide real-time knowledge about your documentation project—such as who is editing what.



- Cursor and selection indicators There are visual cues that display for each user, indicating that multiple people are using the file and other authors are highlighting text selections.
- Filter The Files tree includes a filter button. Use to search project files based on file type or file status.
- New File and Upload File(s) Buttons are located in the main toolbar, one for creating a new file and one for uploading file(s).
- Workspace Overview Click the button to initiate an overview, which gives a report on the workspace. Use to manage your workspace and to identify items, such as files that are not committed, user status, and what needs to be done.
- Status indicators The Files tree shows at-a-glance status. For example, a purple circle next to a file indicates that it has uncommitted changes.
- User status A user's avatar or initials display in the editor's local toolbar when editing a file. Avatars of any other authors working in the file will display next to yours. Your avatar is unique because you can click the drop-down next to it and select your status from the menu.
- Version history When editing a file, a timeline displays with a history of file commits and versions including edits.

States of the Workspace

When you and other authors are editing a file, it is the same file, in a working state, for everyone. That is why you can see changes happening.

- In Progress A file is in a non-committed state and is opened or being edited by authors. The version history indicates the file is in progress and displays the number of edits and users in the workspace.
- Ready to Commit (Optional) A file is done being edited. The file is saved locally, but it is still a working copy and needs to be committed to the repository. Use this state if you are done editing but you notice others are still working on the file. That way you are not committing other authors' changes before they are finished.
- Commit A user or manager can commit a final version of file changes live to the repository. The action of committing (e.g., pressing the Commit button in the dialog) saves the edits and commits changes from all users—not just your changes—to the repository. When a file is committed the user statuses are cleared.

User Status

For every author in a file, you will see each user's avatar with status in the editor's toolbar.

- Avatar (or initials) with a purple circle  This indicates a user is In Progress with a file.
- Avatar (or initials) with ...  The three-dot icon indicates that a user has a file open and is actively editing it. If you click off the current file to open a different file, the icon changes to show that you are In Progress (i.e., a purple circle).
- Avatar (or initials) with a check mark  An author is done making edits and has indicated this by selecting "Ready to Commit." The file is not committed yet. If a commit occurs, the check mark disappears and the version history shows the file as committed.

> NOTE If there are too many users to show each avatar in the toolbar, a three-dot icon indicates more users .

What's Noteworthy?

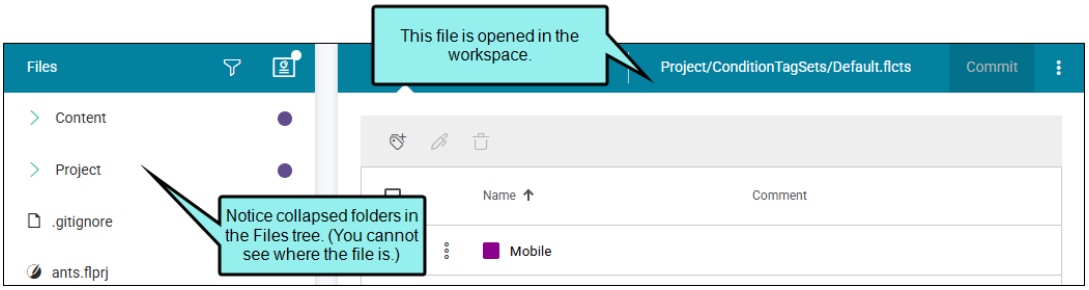
NOTE The workspace vertical three-dot menu includes a couple notable options.

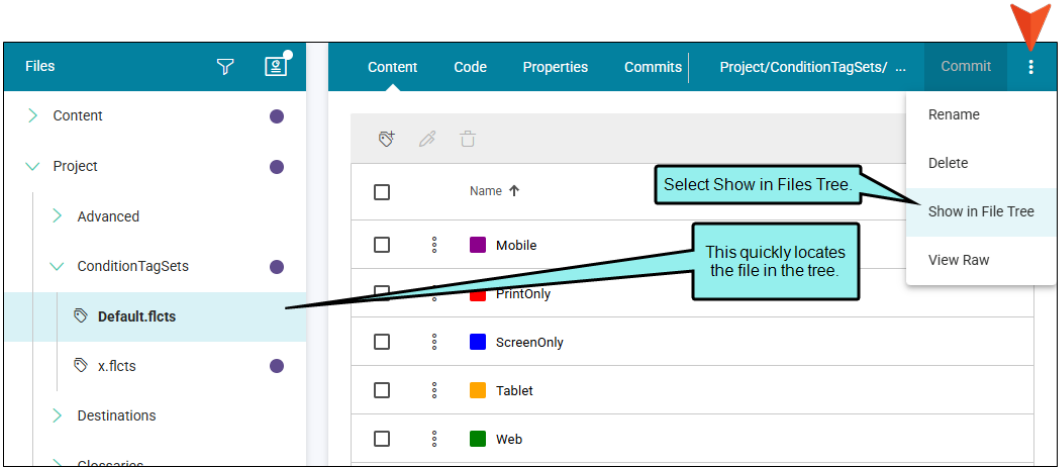
Discard Changes

This allows you to get rid of pending edits without having to commit the file.

Show in File Tree

The workspace vertical three-dot menu includes a Show in File Tree option—evident in the Content Editor for all file types. This feature locates the file in the project Files tree. For example, if folders display collapsed in the workspace Files tree, you can select this option to expand the folders to reveal the location of the file.





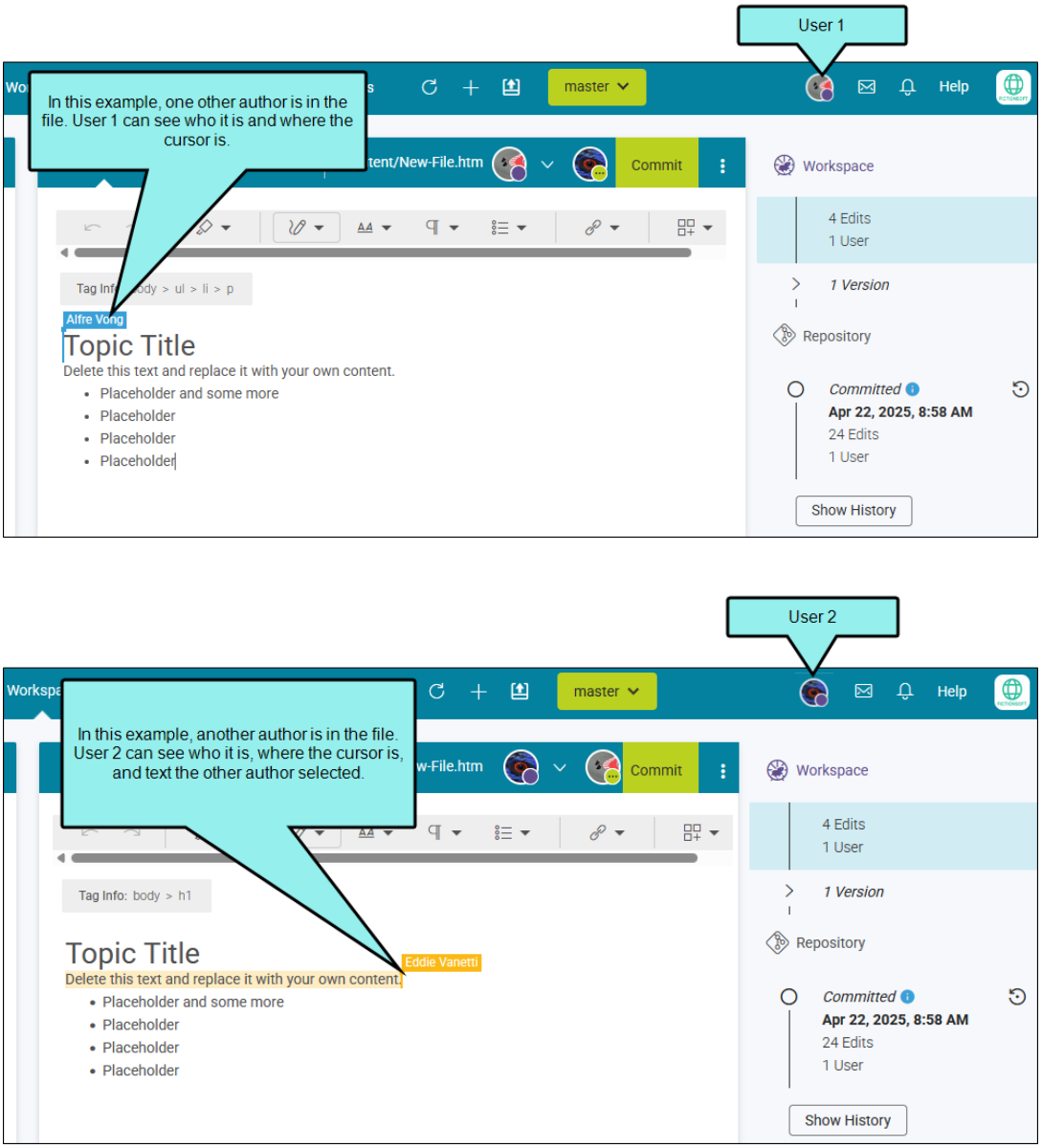
Visible Changes From Others

When a file is selected from the Files tree, it opens in the Content Editor—in a working state.

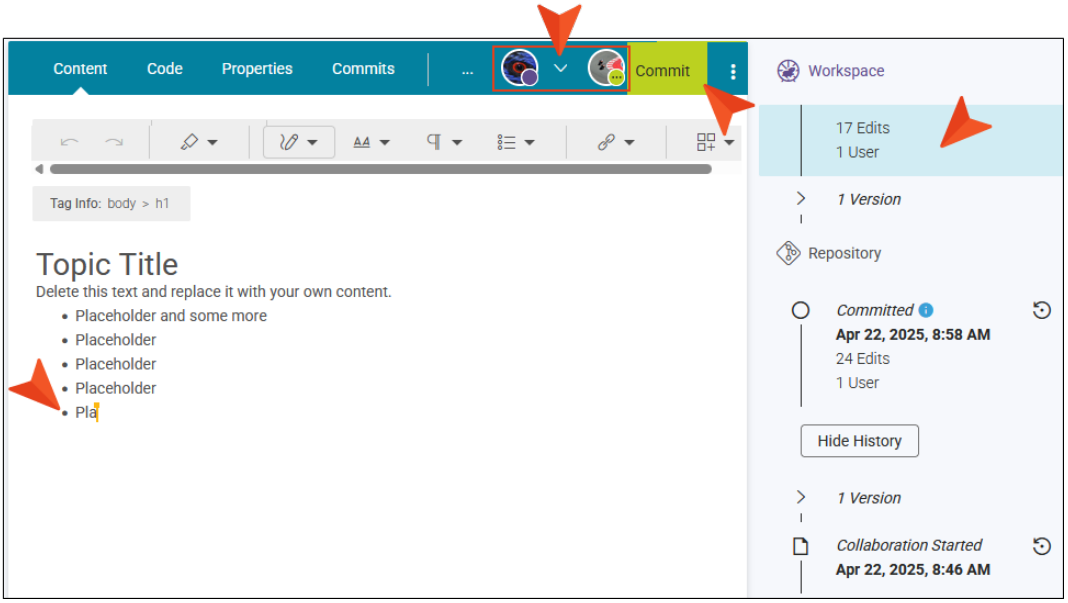
A single user can begin making edits to the file. Once editing occurs, the version history on the right is activated and is updated as edits happen. When the editing is done, the file can be committed into the repository. Pending edits may also be discarded.

If multiple users are using the same file, you can see the changes the other authors are making to it in real-time. There are visual cues that display for each user, indicating that multiple people are using the file and other authors are highlighting text selections.

☆ EXAMPLE Two authors are using the same file in the editor. Notice the cursor and selection functionality that is visible to each author.



As soon as someone starts editing the file, edits are displayed in real-time for each user, the Commit button is enabled, and the version history is activated for the file.



(After a short time, the name associated with a cursor and selection highlights of content disappear to remove unnecessary marks from the workspace.)

Do I Have to Worry About Conflicts?

One of the biggest benefits of collaborative authoring is that you can avoid conflicts that you might otherwise encounter, at least when authors are editing files in the same branch in Flare Online. That's because each author's changes are visible and automatically incorporated when the commit is made. You can collaboratively and concurrently author a file without the worry of losing work or running into conflicts.

Keep in mind, however, that conflicts could still eventually arise later under certain circumstances. For example, you might have made changes to a file in Flare Desktop, and when you do a pull, you might see conflicts with changes made in the same file in Flare Online. Another example is when you use Flare Desktop to merge files from one branch into another; if changes are different in the same file, you might need to resolve conflicts.

> NOTE Flare Online is a Git-based system, where files are version controlled and tracked using the system's source control. The system implements "rules" for resolving possible conflicts. The platform allows for secure co-authoring from anywhere with any device, and accessibility to your online files from the Git repository with no memory or performance impacts.

Main Activities for Collaborative Authoring

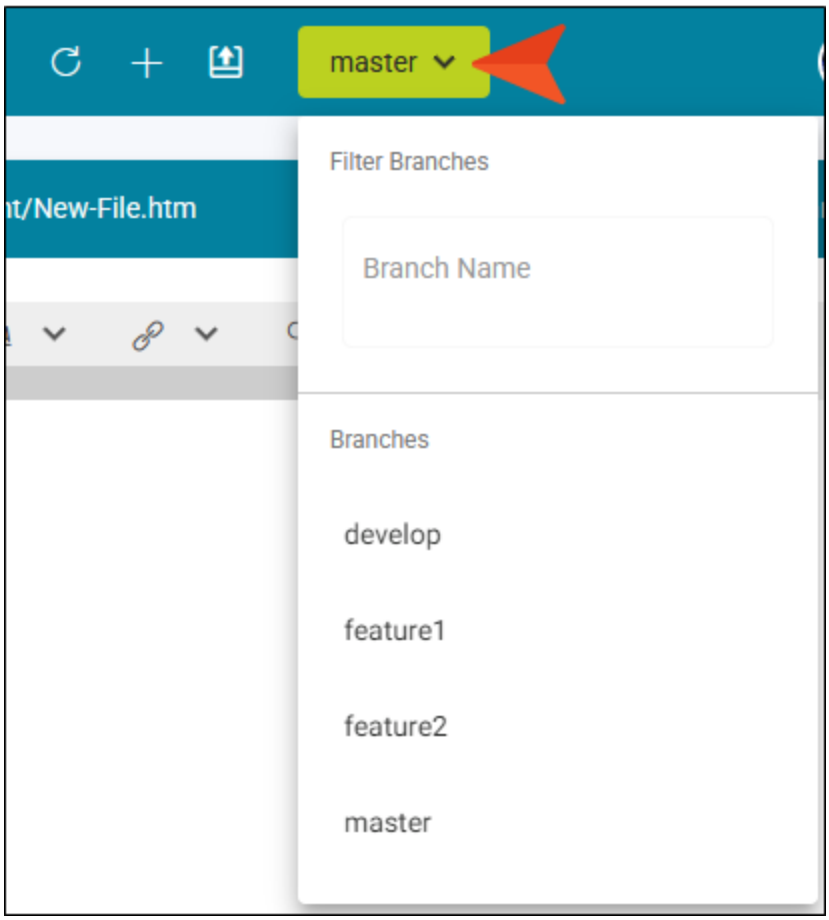
Some activities are particularly common and important when it comes to this feature.

Editing With Multiple Authors

For multiple authors editing the same file at the same time, the following workflow serves as a guide.

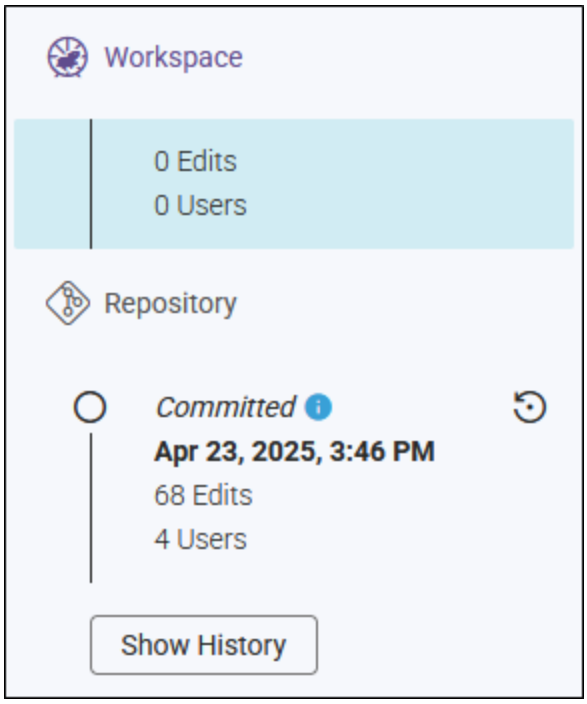
Multiple Authors - How to Edit an Existing File

1. On the left side of the Flare Online interface, click Projects.
2. Select a project to open it.
3. Click the Workspace tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.

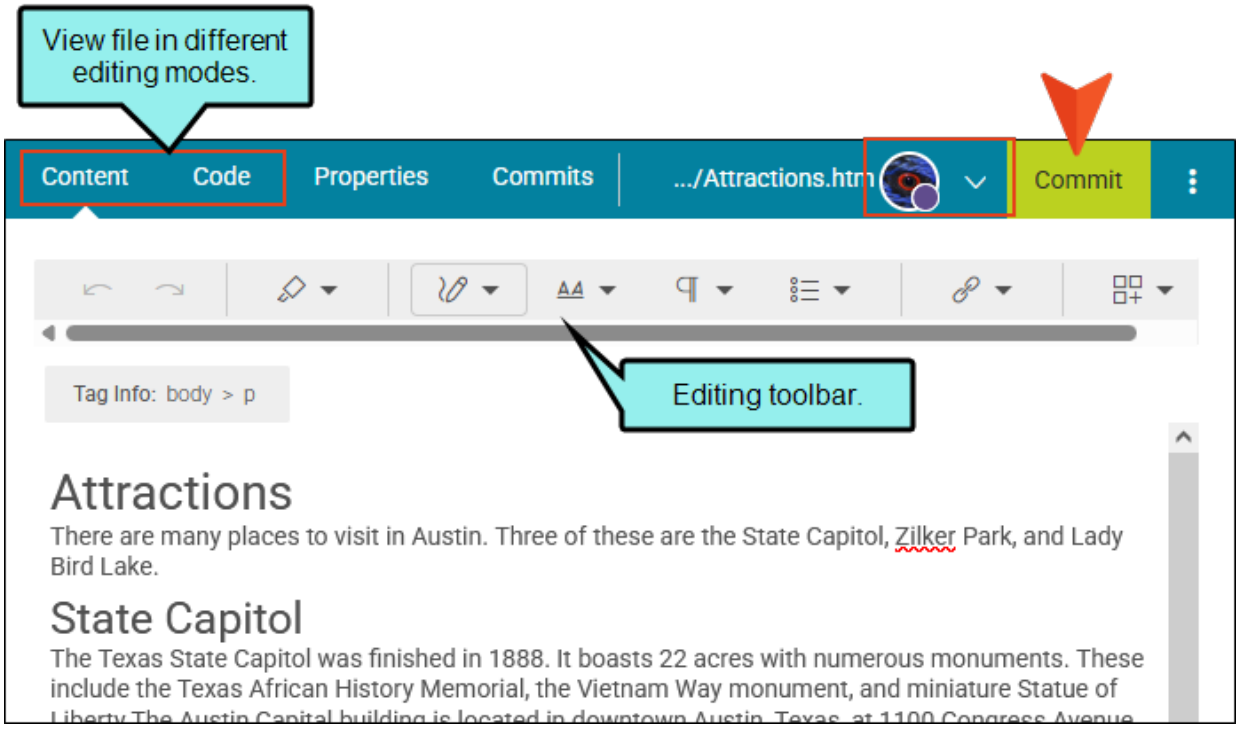


5. (Optional) Click . The Workspace Overview opens to display various items (e.g., status, authors, type) about edited files in a project. (If files are already in a working state, the Open Workspace Overview button shows with a circle in the upper-right corner .)
6. From the left side of the page, expand the existing folders to navigate to a file. You can also click to search for a specific file.

7. Select a file. It displays in the editor to the right. The right gutter switches from showing project activities to displaying a version history for the file.



8. Click in the workspace and make an edit (this activates the file in an uncommitted state for editing, auto-sets your status to In Progress, enables the Commit button, and begins to populate version history). Use the toolbar at the top of the editor to manage the content. Also, the tabs at the top of the editor (i.e., Content, Code, and Commits), allow you to switch modes so you can edit the content or markup, and view commit details.



NOTE The tracking changes option is off by default for collaborative authoring. From the Content Editor's toolbar, click **Review** drop-down > **Track Changes** to enable or disable the feature.

9. When you are done editing a file, it needs to be committed into the Git repository. You can look at the local toolbar or the Workspace Overview to see the status of others.

   Do one of the following:

   - If you determine that you are the last one to finish editing the file, then it is safe to commit the file. From the upper-right corner of the workspace, click **Commit**. (The Commit dialog will be free of a warning about other users in the file.)

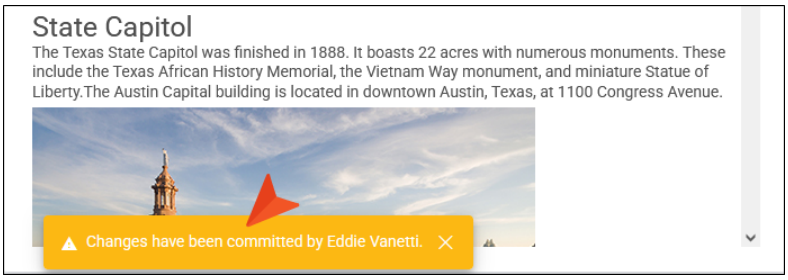
     Alternatively, you can use the Workspace Overview to perform the commit operation. When a file or many files are in a Ready to Commit state, you can commit them one at a time or in bulk.

   - If other authors are still editing, you can let others know that you are finished with the file. From the local toolbar, select the drop-down by your avatar and click **Ready to Commit**.

     When all authors have indicated they are done and the file is ready, then any author or manager can commit the file. If this is the case, click **Commit**.

     > NOTE If multiple files are In Progress, but only one is marked as Ready to Commit, the Commit button is active. This is because you do not have to wait until all the files are complete before committing them. For example, an author goes on vacation but the In Progress state for the file needs to be committed before the author returns.

     > NOTE If other authors are still working on a file and their statuses indicate In Progress, you can still perform a commit (although you should probably communicate with them first). If you do commit a file while other users are in this state, they will see a notification in their editor.

NOTE You can quickly get rid of pending edits in the workspace without having to commit the file.

10. In the Create New Commit dialog, verify the new file path, and enter a **Commit Message**.
11. Click **Commit**.

NOTE With multiple authors, it is the user's responsibility to be aware of other people editing the same file.

NOTE If an author is editing a file, but never commits the file to the repository, another user (author or manager) can commit the file at any time.

NOTE While you are editing a file, an external commit from another source might occur. If this happens, it is best to manually pull the changes into your workspace so you are working with the latest updates.

NOTE If you are collaboratively editing with snippets and changes are made from an external source, the pull bar and Pull Changes button displays in the Snippet dialog.

Committing Edits

Why do I have to commit edits? If you open and view a file, it is in a committed state in the project's repository. If you click to edit a file, it then enters a "workspace mode," essentially making a working copy of the file from the repository. In this mode, the file is in a non-committed state with pending changes. When editing is completed, the file needs to be committed back to the project's repository.

Guidelines for Committing Edits

In general, be aware of the following when committing edits:

- Single file If you are editing a file and click Commit, you are committing what you are looking at in the editor. It can contain edits just from you (one user), or edits from several authors in the same file. (You can also commit a single file via the Workspace Overview.)
- Multiple files Instead of committing a single file, you can bulk commit multiple files at once. Note that many files can be in an uncommitted state which might contain edits from you and other authors.
- Navigating to other files When you work in a file, you can move to another file and work in that, then move to a third file and make changes, and so on. In other words, you do not need to commit an open file before navigating elsewhere. The changes in each file are automatically saved, but they won't be added to the repository until you (or another author) finally perform a commit.
- Committing edits can affect other users Because collaborative authoring is always happening, authors performing a commit are doing so not only for their changes, but also for any other author's changes for the same file. For this reason, it's important to stay in communication with other authors to make sure you are not committing changes that they do not yet wish to be committed. However, the editing process includes safeguards and visible cues in the interface to commit files without issues.

How to Commit Edits

Do one of the following:

- In the Content Editor, from the upper-right corner of the workspace, click Commit.
- In the Workspace Overview, select a file (i.e., the check box next to the file in the grid) and from the upper-right corner click Commit. Alternatively, click the vertical three-dot icon next to the file and select Commit. You can bulk commit files with edits.

What's Noteworthy?

NOTE It is a best practice to simultaneously commit all related files based on the changes you are making. For example if you edited four files to complete a changed feature, then commit all four files together when your changes are done. This ensures a completed collective version of your work in the revision history that you can view and revert back to.

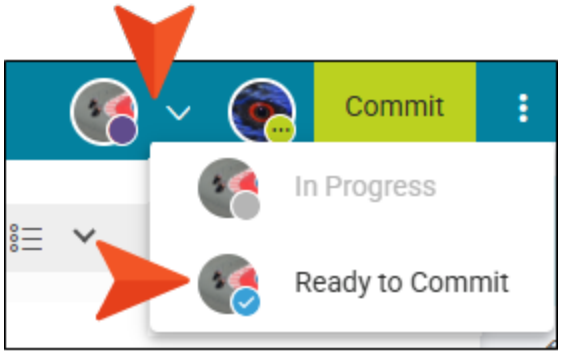
NOTE Until commits are made, changes from those files will not be included in any builds that are generated.

NOTE You can quickly get rid of pending edits in the workspace without having to commit the file.

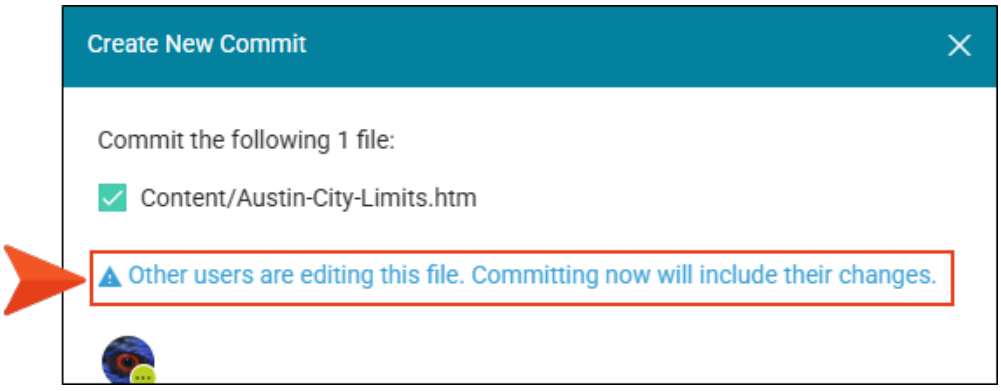
Selecting Ready to Commit

What if you are finished making changes in a file and do not plan to make any more, but other authors are still working in that file? If you perform a commit, you are going to commit their changes as well, and they might not want that.

Fortunately, there is an alternative, you can click your small avatar at the top of the file being edited, and from the drop-down you can select **Ready to Commit**. This is simply a way of signaling to others that they are free to perform a commit for your changes as well as theirs.

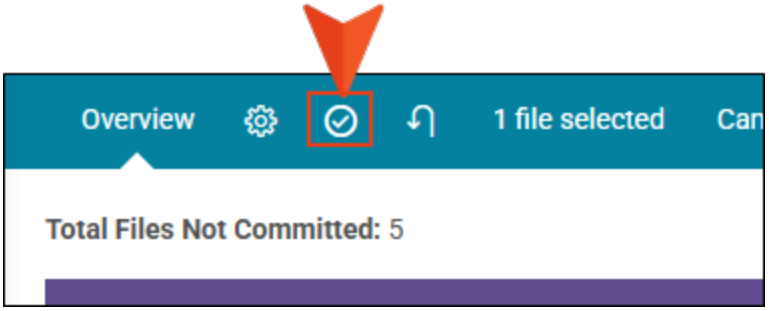


If you do happen to click the Commit button when there are still uncommitted changes from others in the project, you will see a message warning you of this. (If other users have all selected Ready to Commit then you will not see the warning.)



Of course, you can ignore the warning and continue with the commit, but this will commit changes for others at the same time.

NOTE If you set the status to Ready to Commit, but you realize you have more changes to make, you can update the status again. From the drop-down next to your avatar, select In Progress. Or, from the Workspace Overview, select the file, and click the Change My File Status button to update it.

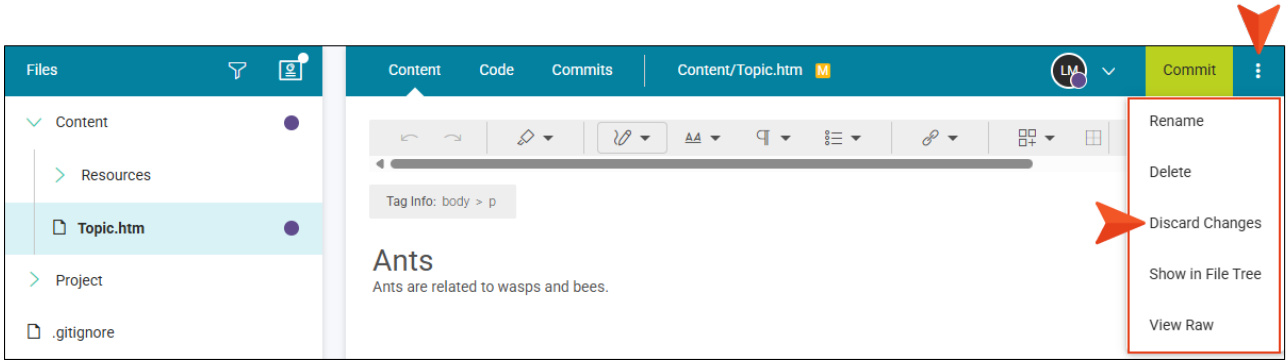


Other Activities for Collaborative Authoring

In addition to the main activities, there are some other tasks you might perform regarding this feature.

Discard Changes From the Workspace

You can discard changes in the workspace (i.e., delete or clear pending edits) without having to commit the file.



Why Is Discarding Changes Significant?

Being able to discard changes is significant because you can remove pending edits from the workspace—to the point of restoring the file to its latest committed state—without having to commit the file.

You can navigate away from the topic and still discard changes. When editing, you could use the Undo button from the local toolbar to reverse the most recent action and step-back edits. But there are limitations to this function. If you move away from editing a topic by clicking another topic and then navigate back to the topic you were editing, the Undo button resets (i.e., it is disabled until new editing occurs). Instead, you can simply discard the changes.

You can quickly remove edits and start over. You could revert to a previous version, but you would still need to go through the process of committing a file to the repository. This might seem arduous if you just want a fresh start to editing a topic.

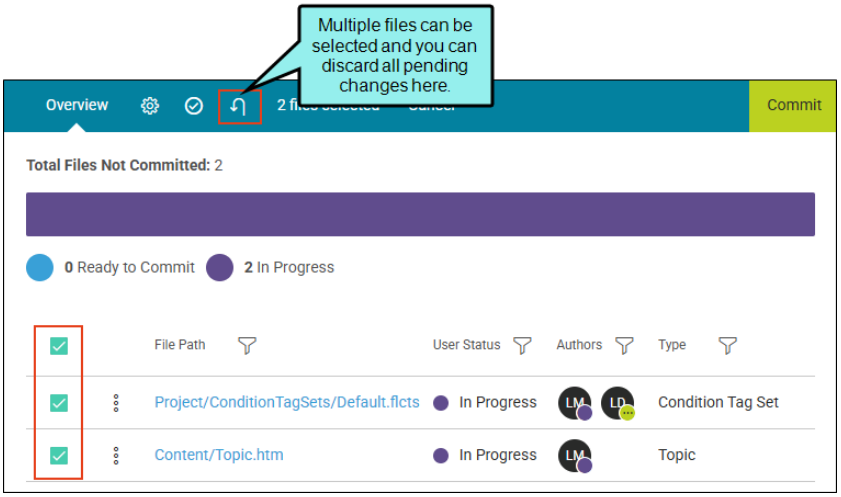
NOTE If an author has pending edits in a topic and decides to revert, the revert function requires a commit or a discard of changes to be done.

How to Discard Changes

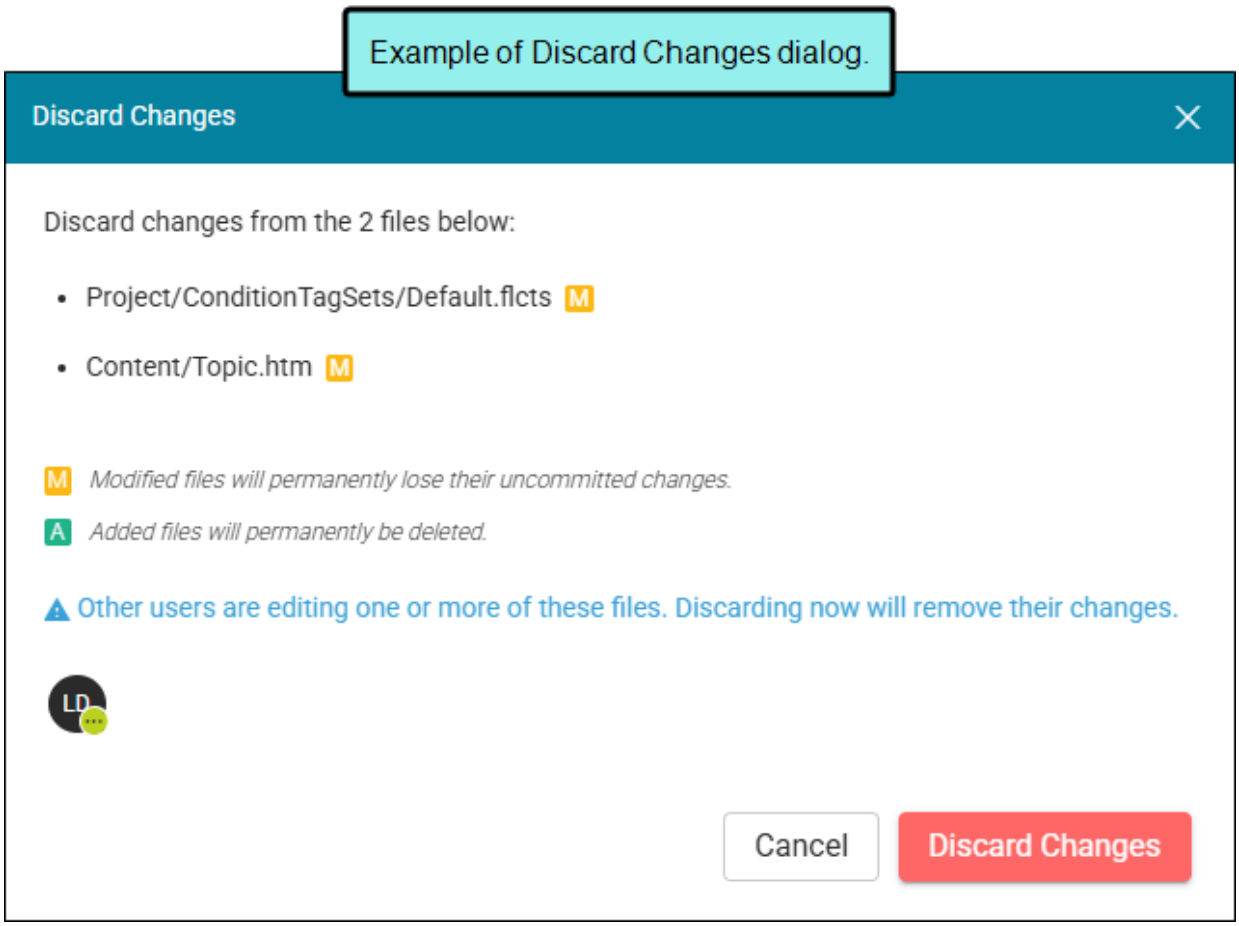
What if I made changes while editing, but I do not want to commit them or add them to the version history? Fortunately, there is a fail-safe option to discard unwanted changes.

1. Do one of the following:
   - From the Content Editor's local toolbar, click the vertical three-dot menu, and select **Discard Changes**.
   - From the Files tree, right-click the file, and click **Discard Changes**.
   - From the file row on the Overview page, click the vertical three-dot menu and select **Discard Changes**.
   - From the Overview page, select a file and click .

NOTE You can discard changes in bulk (i.e., more than one file at a time). From the Overview page select the file(s) and from the local toolbar click .



2. In the Discard Changes dialog, you are alerted to affected file(s). Click the **Discard Changes** button (or click **Cancel** to back out of the operation). If multiple users are editing the same file, the Discard Changes dialog warns you that others are editing and that discarding changes will remove their changes as well.



When done, all pending edits since the last commit are removed from the editor, the Files tree status indicator is cleared, and the file is removed from the Overview page. This means there is no version history for the discarded edits.

NOTE If you click the Discard Changes option, any added file(s) will be permanently deleted, and modified file(s) will permanently lose uncommitted changes.

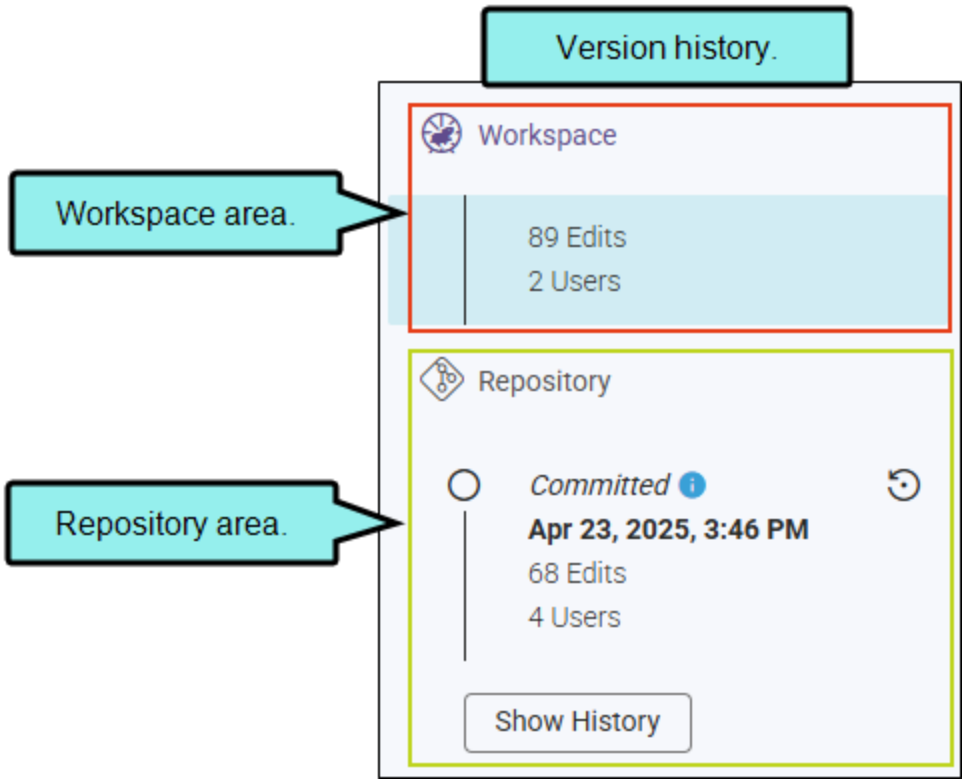
NOTE Discarding changes does not affect the timeline. In other words, when you discard changes, you are only removing edits up to the last commit. The integrity of the version history and the existing commits in the timeline are left alone.

NOTE Discarding changes can affect other users. Because collaborative authoring is always happening, authors discarding changes are doing so not only for their changes, but also for any other author's changes for the same file. For this reason, it's important to stay in communication with other authors to make sure you are not deleting edits that they may wish to keep. However, the editing process includes safeguards and visible cues in the interface to commit files without issues.

Using Version History

The version history area showcases more benefits of collaborative authoring. It maintains a revision history of changes in the project's repository where you can access commit details, view previous versions, and restore a version if need be. It also displays a count of edits and users working on a particular file in the workspace.

The major sections of version history are the workspace and the repository areas. The workspace tells you about an uncommitted working copy of the file. The repository tells you about the committed versions of the file in the repository. The version history for a certain file can get quite long depending on the amount of work applied to it.

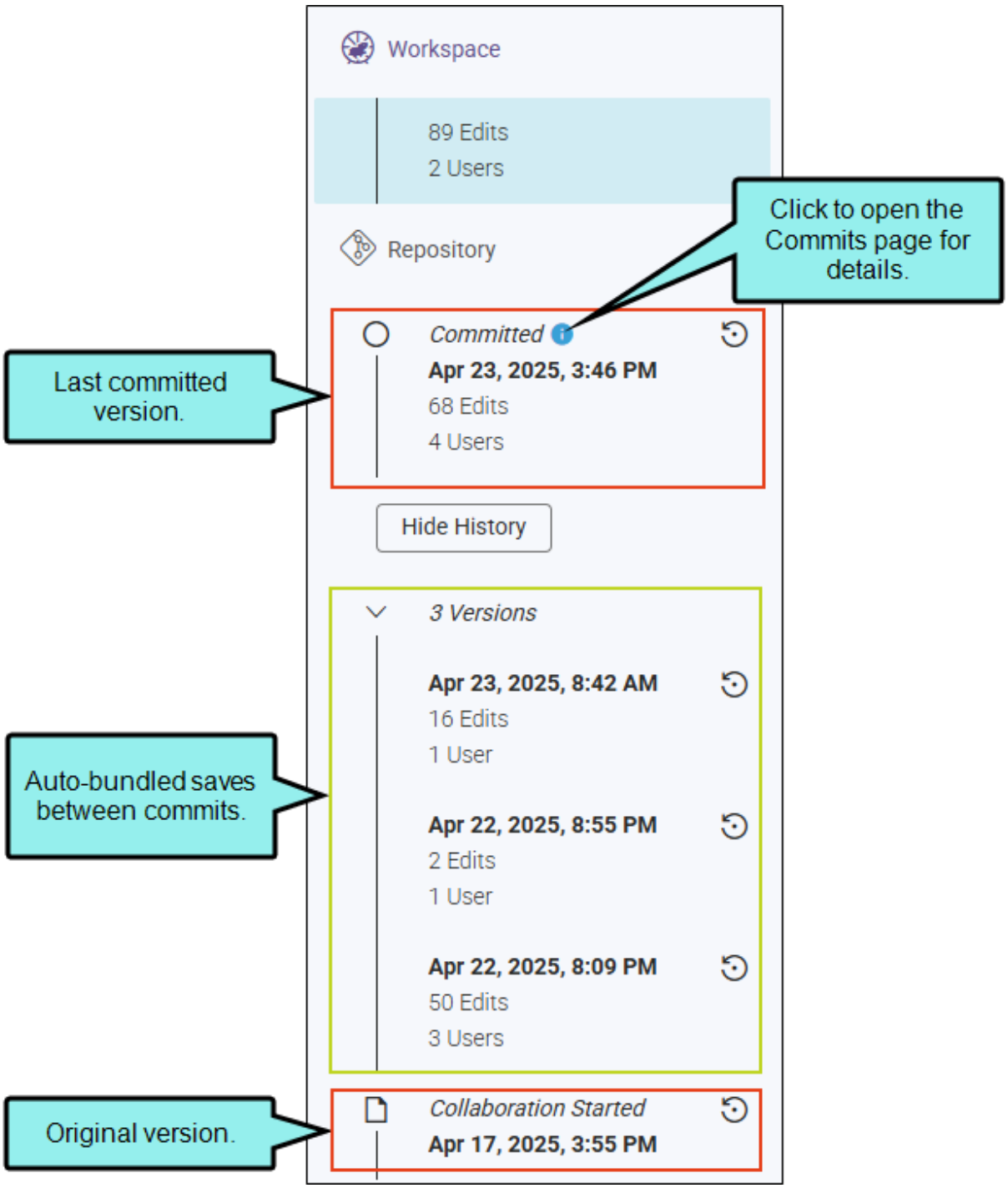


Following are a few things to know about the version history:

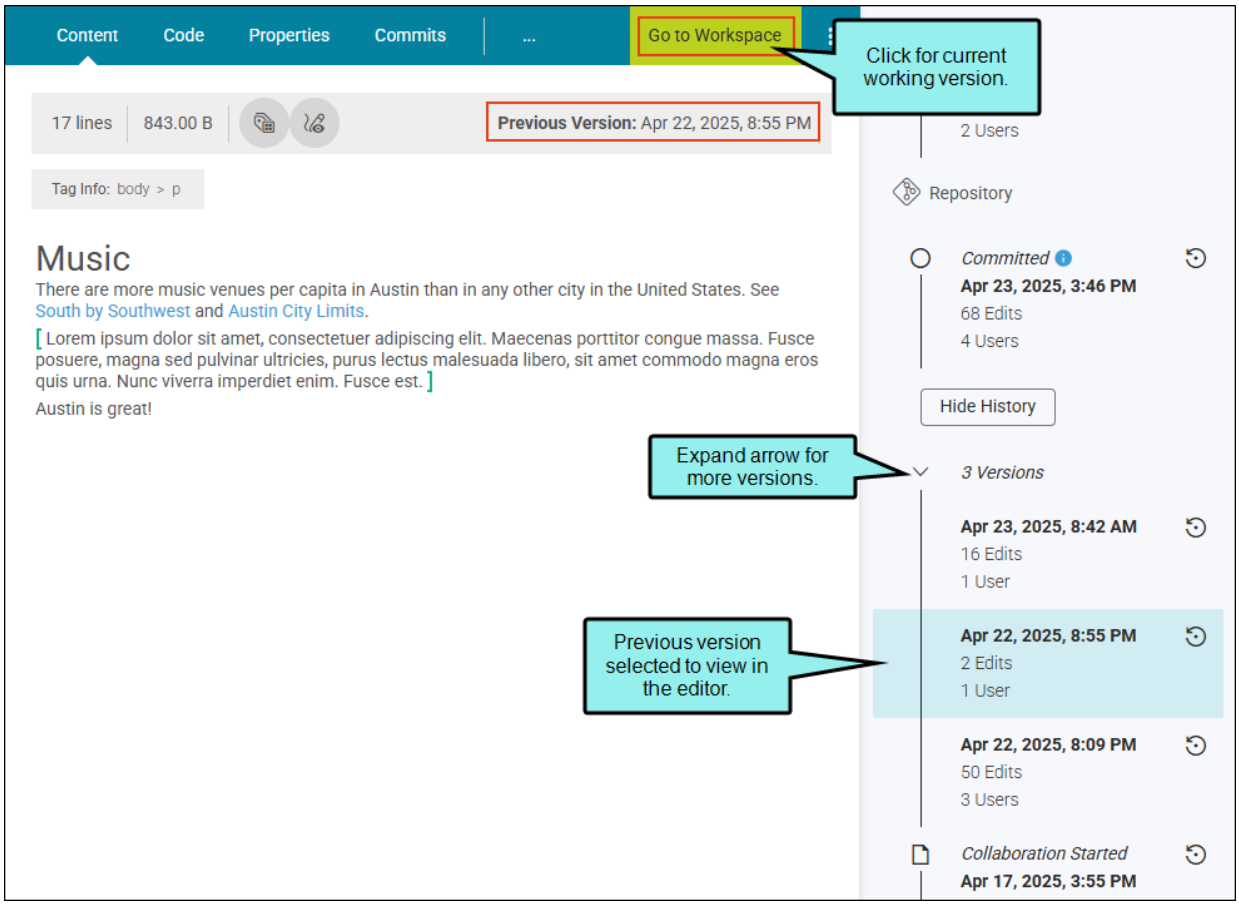
- The last committed version always displays in the repository area. When the history is expanded it shows from the end, not the beginning. In other words, the latest version displays first at the top of the list, and older versions move down in the list.
- The edits number can get high (e.g., 254 edits). Edits are counted per action (i.e., roughly one character change equals one edit). The system controls the count of edits by auto-bundling changes (i.e., creating a working version of the file in the repository) between commits, every 30 minutes.
- The version history is updated based on the branch; and there is only one workspace per branch.

How to View a Previous Version

1. On the left side of the Flare Online interface, click Projects.
2. Select a project to open it.
3. Click the Workspace tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.
5. From the left side of the page, select a file. The last committed version displays in the editor, and the version history area shows on the right side of the screen.
6. In the version history area, click Show History. Upon selection the button changes to Hide History—so you can collapse the history. If revisions exist for the file, the timeline expands to show the versions (i.e., original when it was created, auto-bundled saves between commits, and committed files.)


Workspace
89 Edits
2 Users
Repository
Committed
Apr 23, 2025, 3:46 PM
68 Edits
4 Users
Hide History
3 Versions
Apr 23, 2025, 8:42 AM
16 Edits
1 User
Apr 22, 2025, 8:55 PM
2 Edits
1 User
Apr 22, 2025, 8:09 PM
50 Edits
3 Users
Collaboration Started
Apr 17, 2025, 3:55 PM
Click to open the Commits page for details.
Last committed version.
Auto-bundled saves between commits.
Original version.

7. Select a previous version. (To select an auto-bundled version, you might need to click the down arrow to expand the timeline further.)
8. The editor displays the version selected. You might notice older edits.
9. Click Go to Workspace to return to the current working version.



How to Revert to a Previous Version

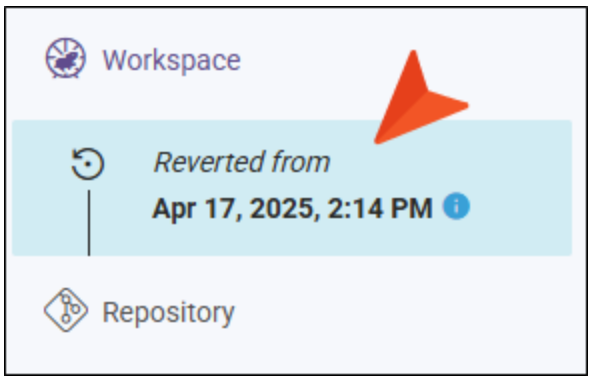
You can revert to a previous version of the file (i.e., to the original version, a committed version, or an auto-bundled version).

1. On the left side of the Flare Online interface, click **Projects**.
2. Select a project to open it.
3. Click the **Workspace** tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.
5. From the left side of the page, select a file. The last committed version displays in the editor, and the version history shows on the right side of the screen.
6. In the version history area, find the instance of the file that you want to restore. (You might need to click Show History or expand auto-bundled versions in the timeline.)

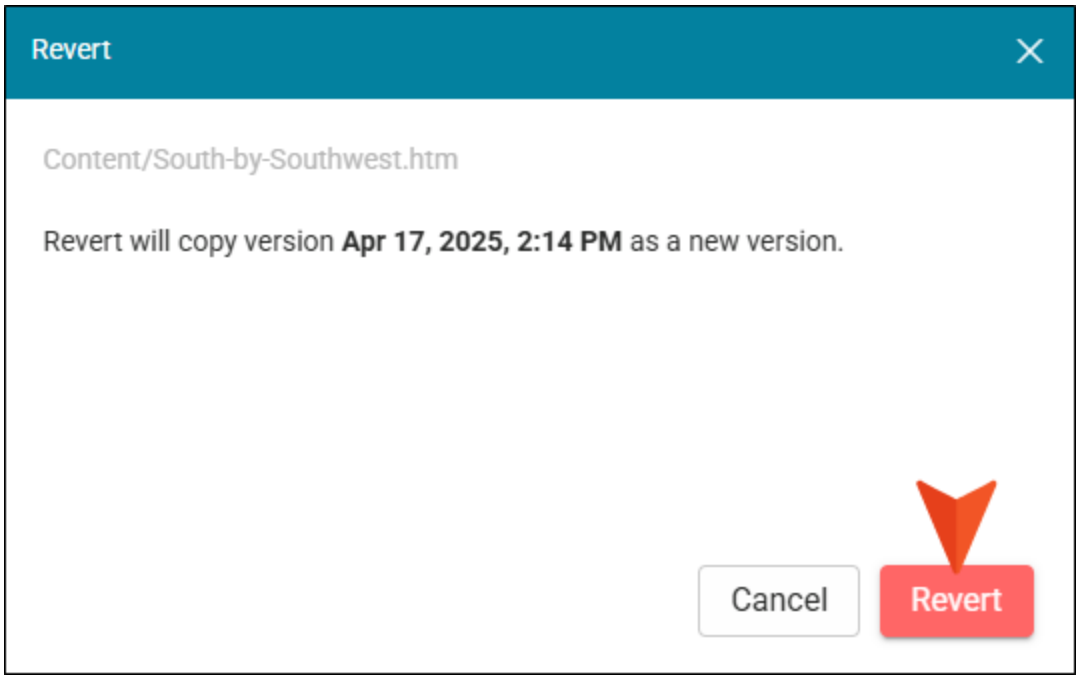
   > **TIP** It might be a good idea to select the version and view it first, before reverting it, just to make sure that version is the one you want.

7. Click **(Revert to this version)**.

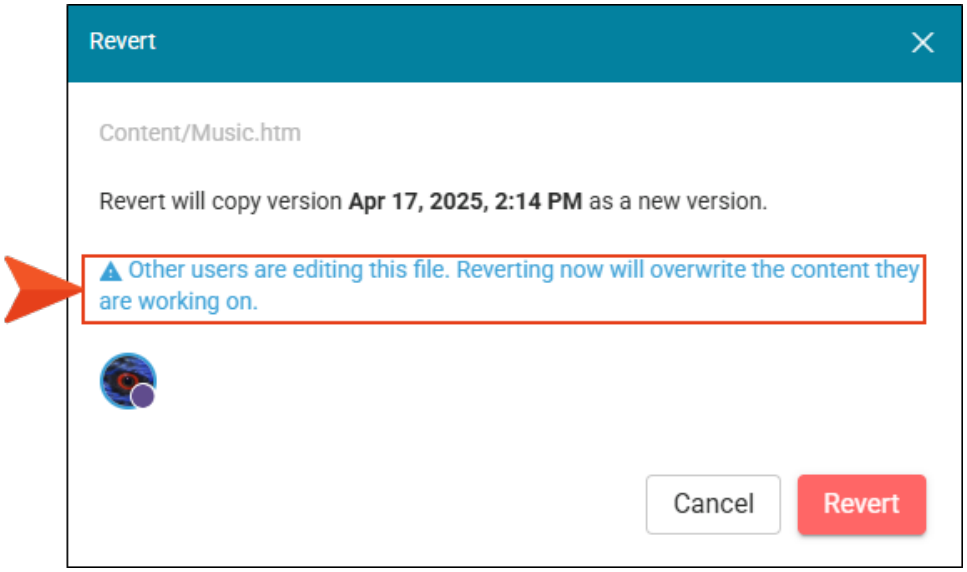
   This action copies the version as a new version, and records it as such in the workspace section.



8. The Revert dialog opens. If no other users are editing the file and it is safe to revert it, click Revert.



NOTE If other users are detected as working in the file that you want to revert, the Revert dialog will warn you.



In this case, you might want to click Cancel. Then, communicate with other team members about the conflict, and decide what action to take that works for your team.

9. Even though the revert updates the workspace version to what it was, it still needs to be committed to the repository. Click Commit in the upper-right corner of the editor.
10. In the Create New Commit dialog, enter a Commit Message.
11. Click Commit. The reverted file displays as the latest committed version in the repository.

NOTE If other authors are editing a file that is reverted, they will see a warning banner display at the bottom of the editor alerting them to the fact. Unless your intent is to overwrite other people's content, it is best to determine the status of the workspace and of other users to avoid problems.

EXAMPLE Multiple users are collaboratively editing a file. One user accidentally deletes all the contents of the topic (and the other users notice deleted text in real-time as they work). The team communicates about the issue and all agree to revert to a previous version of the file. The advantage to this feature is that you can select a previously committed file or select one of the auto-bundled versions to restore. One of the authors is able to identify the version where all the content and most recent edits exist, and reverts to it. The others see the reverted file in their own editor and continue editing until they are ready to commit. According to the timeline, the new version of the file displays at the top of the history as the latest version.

Workspace Overview

The Workspace Overview is essentially a report of the project's current collaborative authoring state. Managers and authors alike may find it useful for monitoring the workspace and for identifying items, such as files not committed, user status (e.g., who is working on a file or who has completed a file), and what needs to be done.

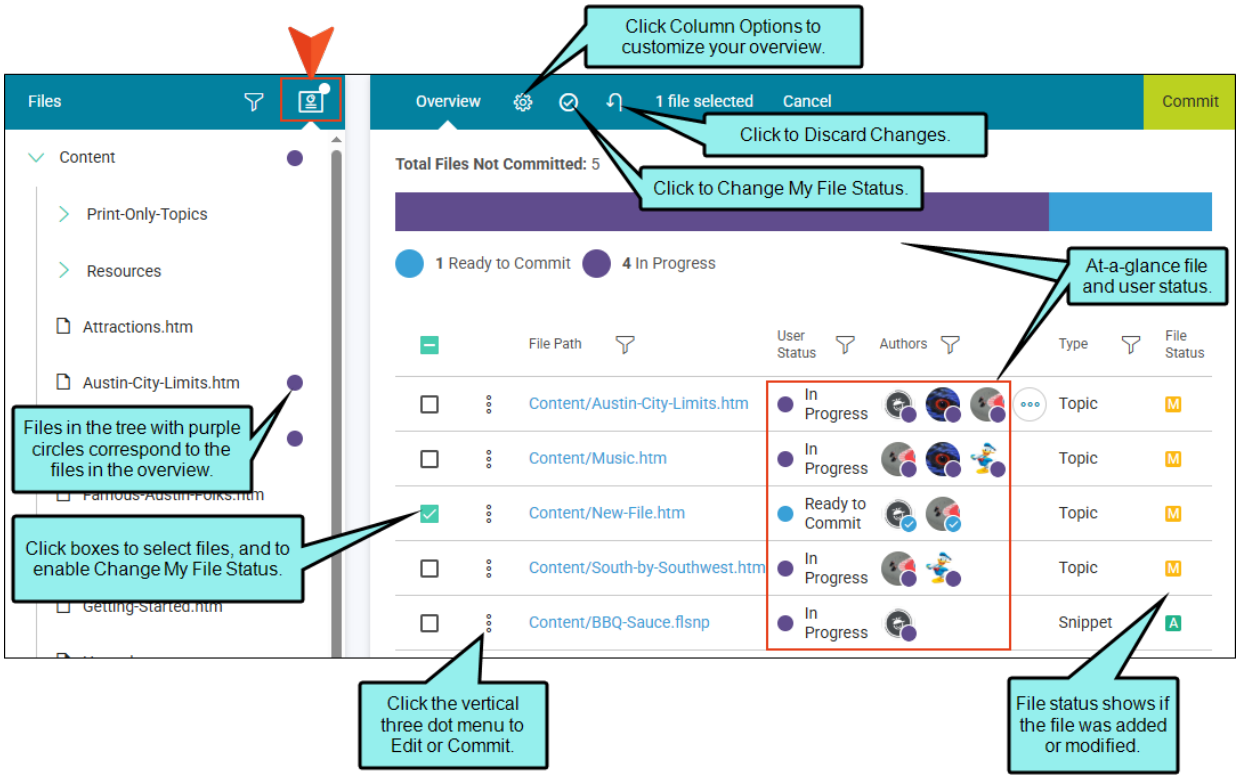
You can customize the columns that display in the overview and perform certain tasks like changing the file status, launching the Content Editor from a file listed, pulling external changes, or committing a file. You can also commit many files at once in bulk, to the Git repository.

How to Open the Workspace Overview

1. On the left side of the Flare Online interface, click Projects.
2. Select a project to open it.
3. Click the Workspace tab at the top of the screen.
4. (Optional) From the drop-down at the top of the interface, you can select a branch for the edits.
5. Click . The Workspace Overview opens to display various items (e.g., status, authors, type) about edited files in a project. (If files are already in a working state, the Open Workspace Overview button shows with a circle in the upper-right corner.)

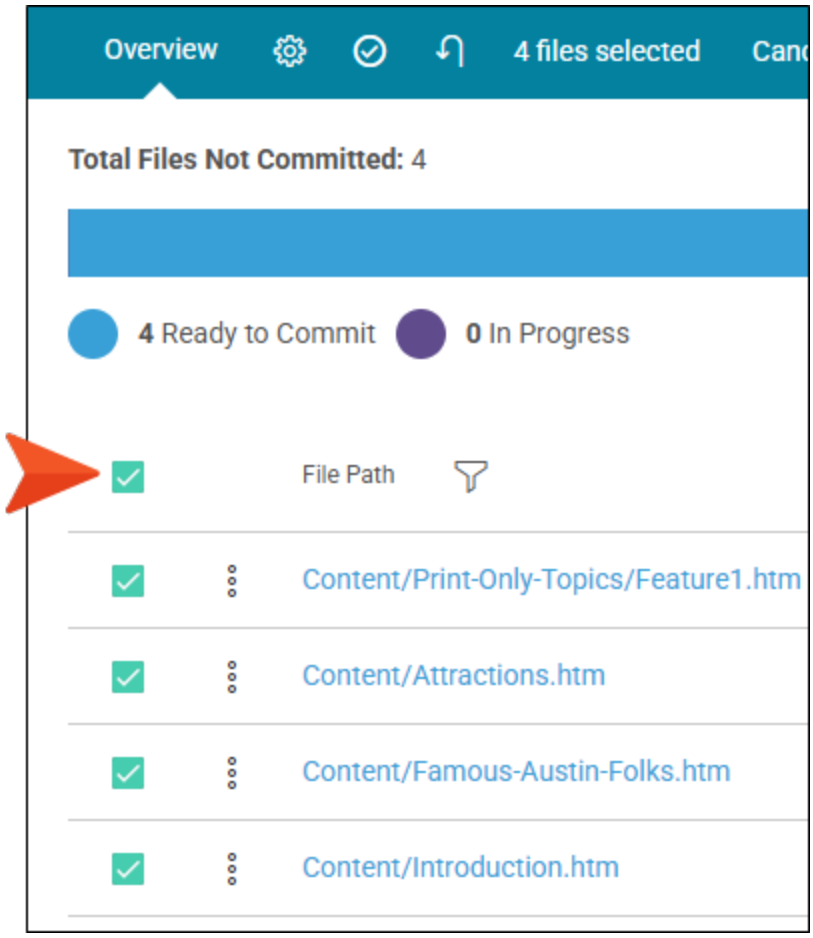
   If there are files uncommitted, you will see a snapshot of the total files not committed (i.e., files that are either In Progress, or files Ready to Commit).

   If all files are committed (i.e., no files are being edited), the overview grid is empty.


Click Column Options to customize your overview.
Files
Overview
1 file selected
Cancel
Commit
Click to Discard Changes.
Click to Change My File Status.
Content
Print-Only-Topics
Resources
Attractions.htm
Austin-City-Limits.htm
Total Files Not Committed: 5
1 Ready to Commit
4 In Progress
At-a-glance file and user status.
File Path
User Status
Authors
Type
File Status
Content/Austin-City-Limits.htm
In Progress
Topic
Content/Music.htm
In Progress
Topic
Content/New-File.htm
Ready to Commit
Topic
Content/South-by-Southwest.htm
In Progress
Topic
Content/BBQ-Sauce.flsnp
In Progress
Snippet
Files in the tree with purple circles correspond to the files in the overview.
Click boxes to select files, and to enable Change My File Status.
Click the vertical three dot menu to Edit or Commit.
File status shows if the file was added or modified.

How to Commit Multiple Files at Once

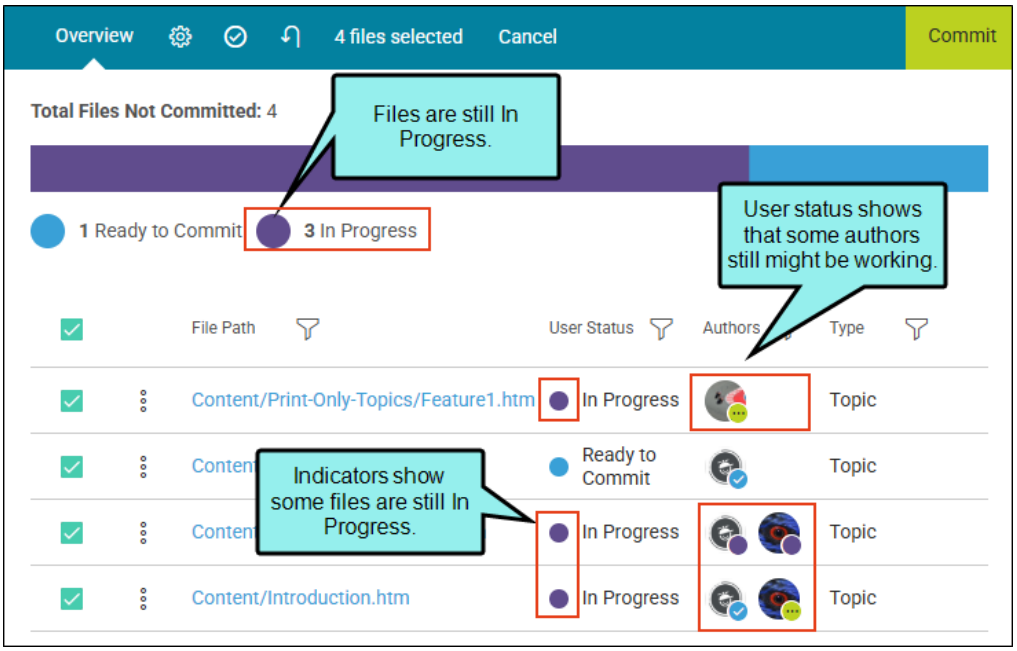
1. In the Workspace Overview, click the check box column header. This selects all the files in the list (for a bulk commit). You can also deselect a few of the files if you want to commit most of files, but not all of them.



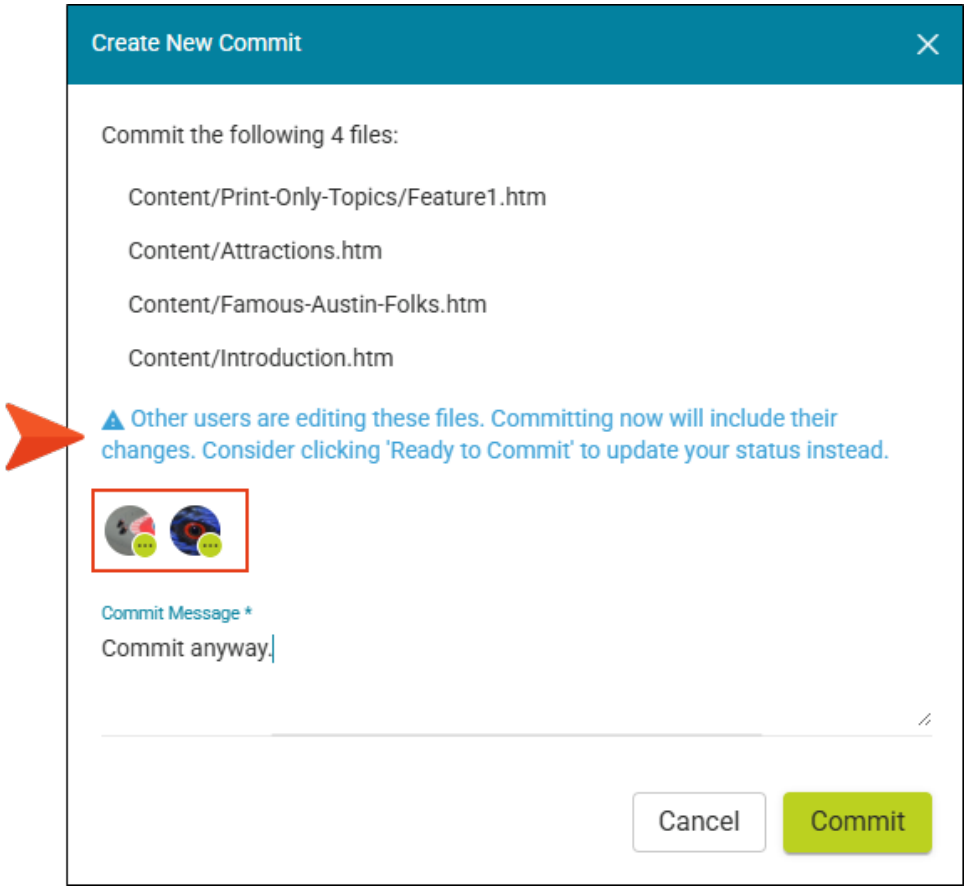
2. Click **Commit** in the upper-right corner.
3. In the Create New Commit dialog, enter a **Commit Message** and click **Commit**. Alternatively, you can click **Cancel**.

EXAMPLE Before you do a bulk commit, use the Workspace Overview to determine if committing the files selected is a good idea. In most cases, authors and files status should indicate "Ready to Commit" for all files. Otherwise, you are going to commit changes that someone else might not want committed yet.

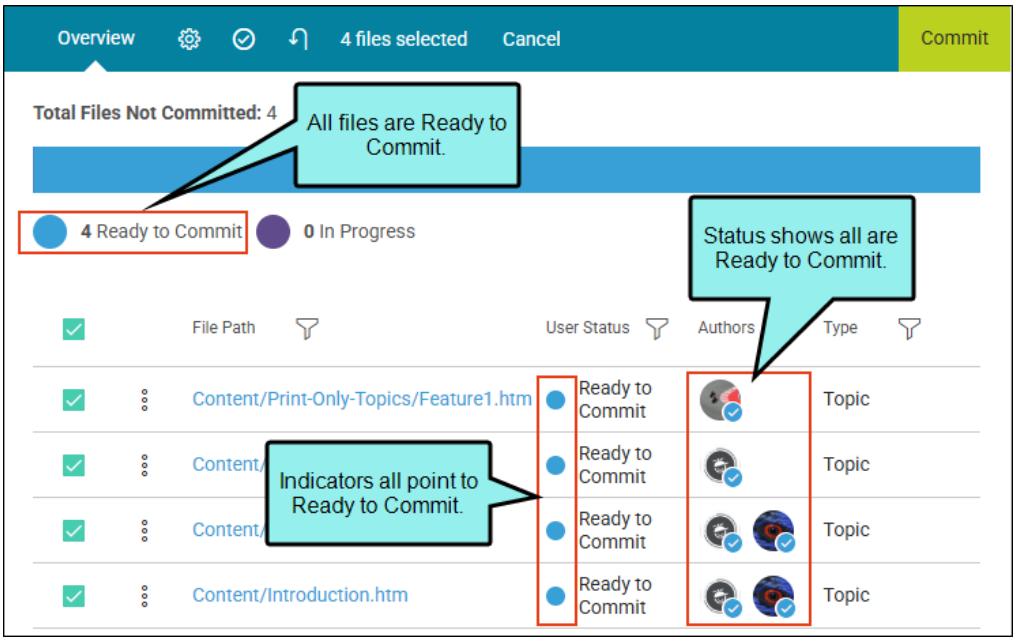
The following Workspace Overview shows an example of files and authors still at work.



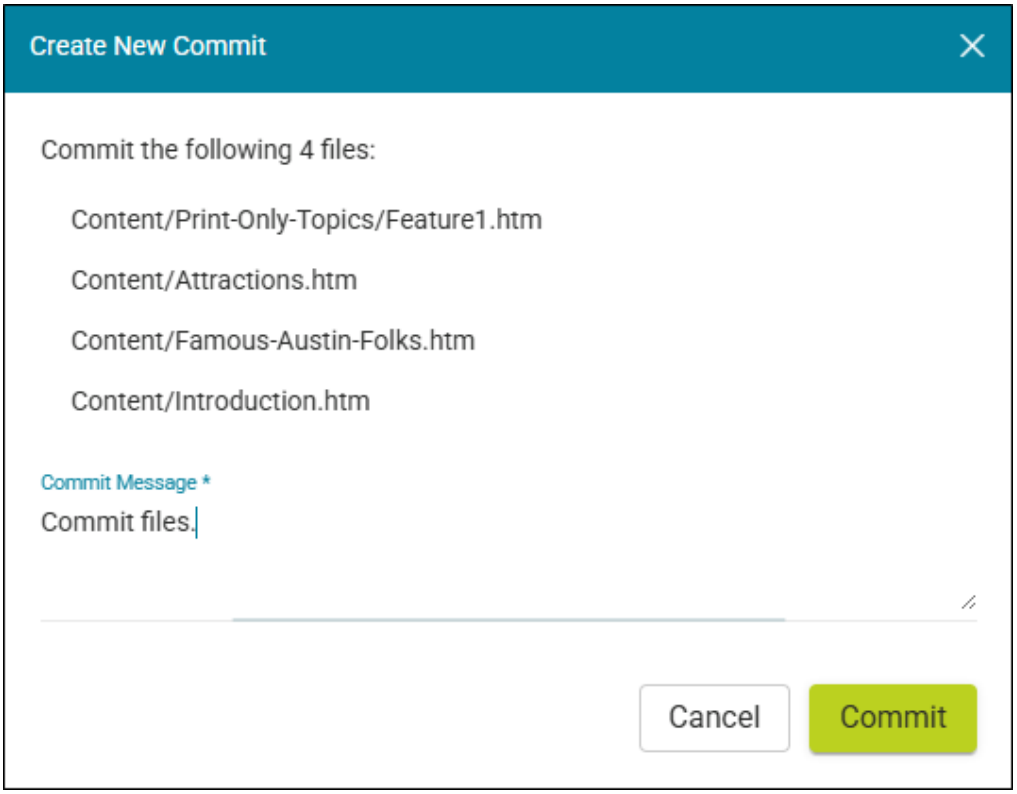
☆ It is probably not a good idea to commit these items yet. If you do, a warning displays in the Commit dialog.



This is a better example. The files and authors are all set for a commit.

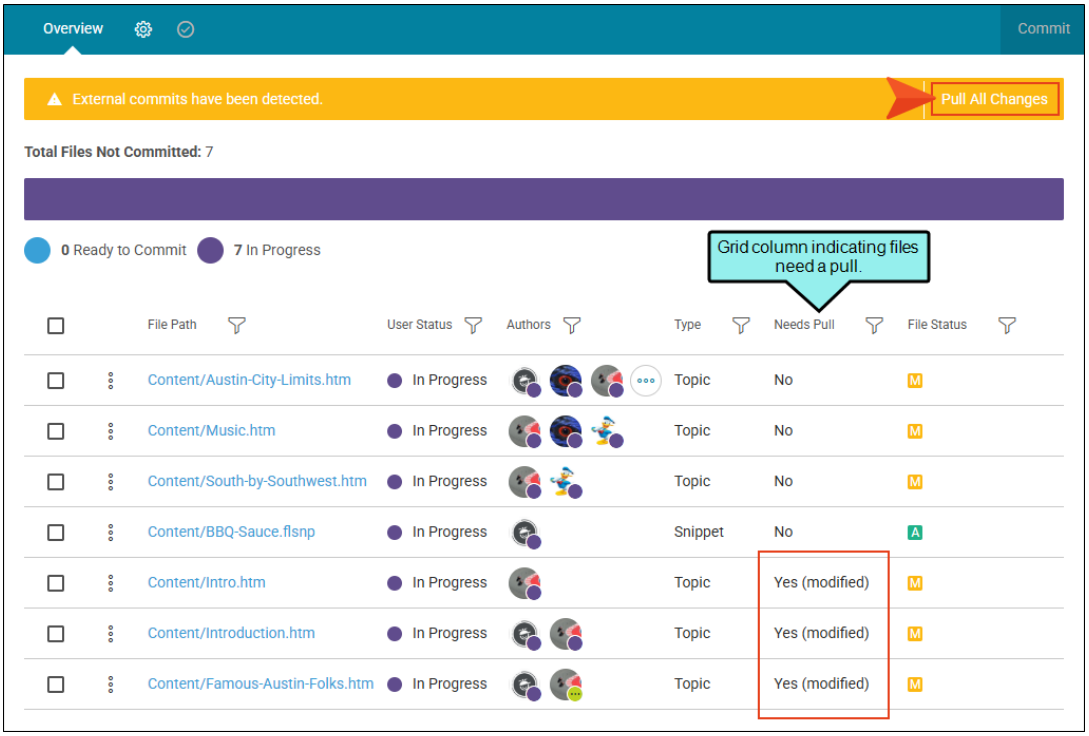


☆ No warning displays in the Commit dialog. And all your co-workers are probably happy that their changes made it into the files without a problem and headache.

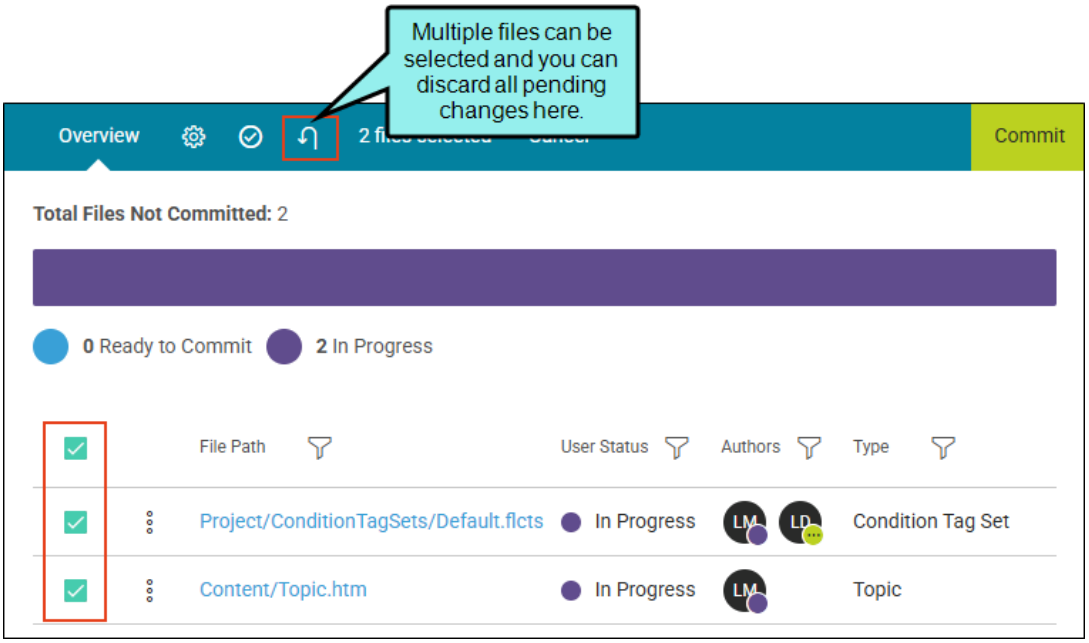


What's Noteworthy?

NOTE If external commits have been detected, a banner displays at the top of the overview, with a Pull All Changes button that you can click to pull those updates. Also, a "Needs Pull" column is available for the grid indicating there are external changes that need to be pulled. Clicking the button from the banner in the Workspace Overview pulls changes for all modified files.



NOTE You can discard changes in bulk (i.e., more than one file at a time). From the Overview page select the file(s) and from the local toolbar click .

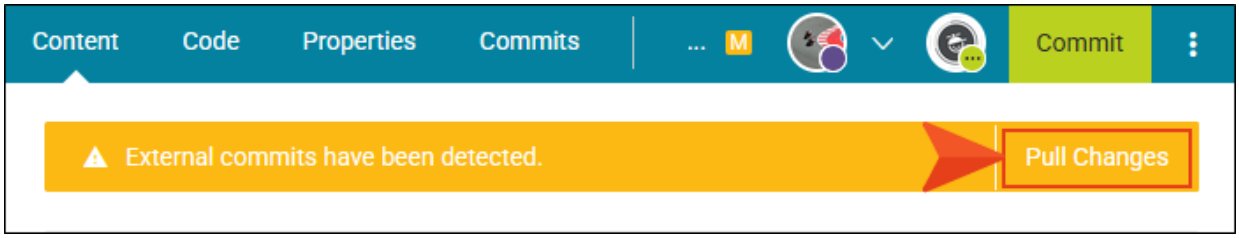


Merging Files With External Commits

Other authors can work in the same file as you using an external source, such as Flare Desktop. Since Flare Online uses collaborative real-time technology it does not lose track of external changes or create conflicts, and you are always guaranteed a valid document. Based on the situation, it will auto-merge changes during a commit, or it will prompt you to manually pull external changes into a working file.

For example, you are working on a file, and someone else edits the same file in an external source. If external changes are made and pushed to Flare Online, a few things can happen:

- If nobody is working on that file in Flare Online, the changes are simply uploaded to Flare Online.
- If someone is working on that file with uncommitted changes, then a pull bar banner displays for the topic with a Pull Changes button. This allows you to manually pull the latest edits into the file without having to commit the file.



- If you do not select to pull external commits from the banner and you try to commit your changes, you will not be able to until a pull is done (and at this point, external commits are automatically merged to the file). You can review the changes from the external commits, and then you can commit your changes.

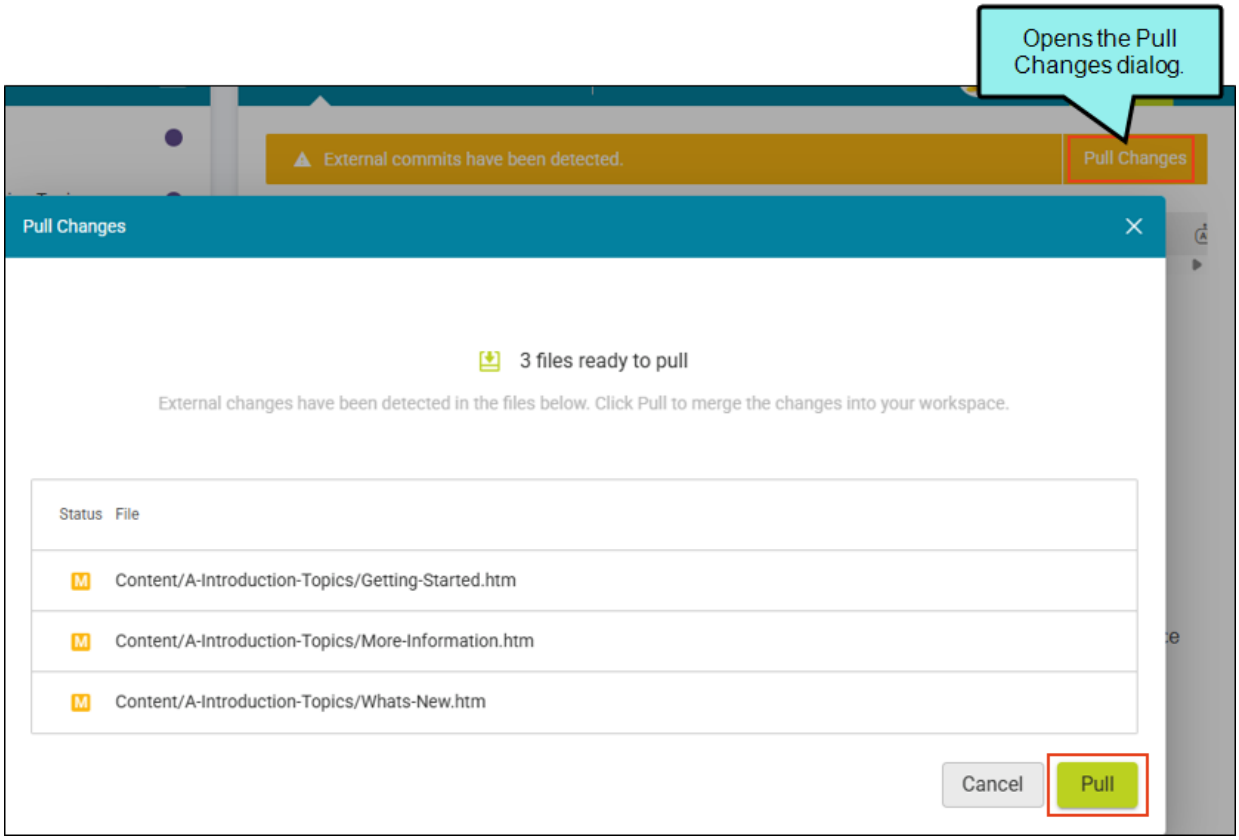
How To Manually Pull External Commits

You can manually pull external commits using the Content Editor or the Workspace Overview (doing it from either place will pull changes for all modified files).

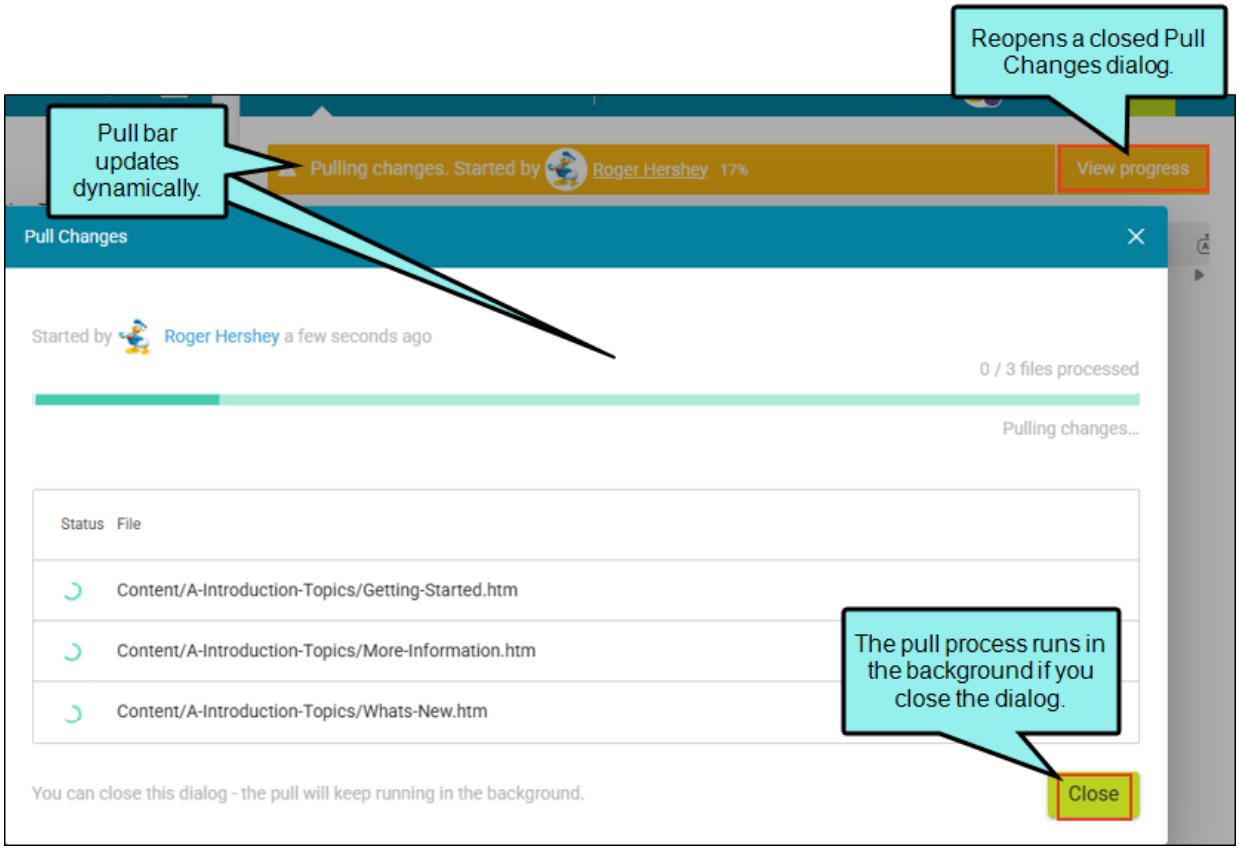
1. From the pull bar at the top of the interface, click **Pull Changes**.



2. The Pull Changes dialog opens. The files are listed in a table that dynamically updates during the pull. Click **Pull** to merge the changes into your workspace.



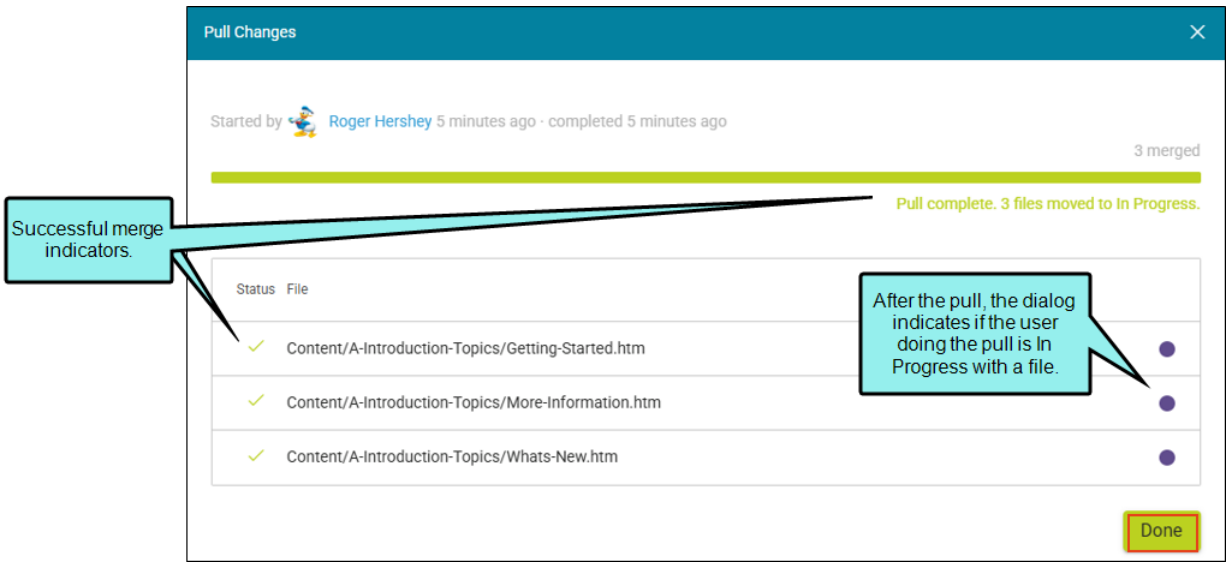
When pulling external commits for multiple files at once (e.g., hundreds of files with many changes), the pull time might take a few minutes. For longer pulls, you can optionally close the dialog. But you can still track the pull process in the workspace (it keeps running in the background). You can click View Progress via the pull bar to reopen the Pull Changes dialog.



You can see real-time merge processing, and information such as who started the pull, and percent complete. In addition, the dialog shows individual file status pertaining to the pull activity. The workflow also guards against multiple pulls at once.

> NOTE Knowing who started the pull is beneficial because if there are multiple users working in the same files online, only one person can do the actual pulling of external changes (and it might not be you). During a pull, all users in the workspace see the same merge processing in the pull bar.

3. When the merge is complete, click **Done**.



NOTE If a file fails to merge, the file will be flagged with a warning status icon and it will also list the pull process issue. If this happens, correct the error and try again.

NOTE Files need to be open, and in an active editing mode in the collaborative workspace for the external commits pull bar banner to display. If not, new changes will just bypass the dialog.

NOTE Flare Online is a Git-based system, where files are version controlled and tracked using the system's source control. The system implements "rules" for resolving possible conflicts. The platform allows for secure co-authoring from anywhere with any device, and accessibility to your online files from the Git repository with no memory or performance impacts.

EXAMPLE The following is an example of how a common merging workflow might play out. A project is bound between Flare Online and Flare Desktop, and two authors are editing the same file, on the same branch, in the same project.

User 1 Open a file in Flare Online's Content Editor and begin editing it.

User 2 Open the same file, but in an external source (e.g., Flare Desktop).

User 2 Edit the file by changing the content, and commit then push any changes.

User 1 A pull bar banner displays at the top of the topic in Flare Online, letting you know that external commits have been detected.

User 1 In the Content Editor, select to pull the changes. When the pull is complete, the changes from the external source are merged into the file.

PDFs

The following PDFs are available for download from the Help system.

AI Assist Guide

Analytics Guide

Authoring Guide

Branding Guide

Building Output Guide

Checklists Guide

Conditions Guide

Getting Started Guide

Images and Multimedia Guide

License Management Guide

Links Guide

Projects Guide

Reports Guide

Reviews Guide

Security Whitepaper

Sites Guide

Snippets Guide

Source Control Guide

Styles Guide

Targets Guide

Tasks Guide

TOC Guide

Topics Guide

Translation Guide

Users and Teams Guide

Variables Guide

What's New Guide

Widgets Guide